IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661, p-ISSN: 2278-8727, Volume 27, Issue 2, Ser. 4 (Mar. – Apr. 2025), PP 69-77
www.iosrjournals.org

Automated Brain Tumor Detection And Classification Using Advanced Machine Learning Techniques

Antony Rex Angeleo A B
Computer Science And Engineering St. Joseph's College Of Engineering Chennai, India

Dhiwakar S
Computer Science And Engineering St. Joseph's College Of Engineering Chennai, India

J. T. Anita Rose
Computer Science And Engineering St. Joseph's College Of Engineering Chennai, India

Abstract
Brain tumors are serious neurological conditions that require early and accurate diagnosis for effective treatment. Conventional diagnostic approaches depend on manual MRI analysis, which can be time-intensive and influenced by human interpretation. This model introduces an automated framework for brain tumor detection and classification using MRI scans. The model applies Convolutional Neural Networks (CNNs) to analyze intricate spatial patterns in MRI images, categorizing tumors into 11 types, including rare ones like Ganglioglioma and Germinoma. The dataset comprises 18,000 annotated MRI scans, ensuring comprehensive tumor representation. The deep learning model achieves an accuracy of 98.7%, outperforming traditional image analysis methods. To improve reliability, data augmentation and transfer learning strategies are implemented. The model is accessible through a Flask-powered web platform, allowing users to submit MRI scans and obtain immediate tumor predictions. The findings highlight that this tumor detection model can enhance clinical decision-making by offering a fast, precise, and non-invasive diagnostic aid for radiologists and healthcare practitioners.
Index Terms: *Brain Tumor Detection, MRI Classification, Deep Learning, CNN, Medical Image Processing, Automated Diagnosis.*

Date of Submission: 03-04-2025 Date of Acceptance: 13-04-2025

I. Introduction

Medical image interpretation is one of the fastest-evolving fields in artificial intelligence, with advancements in machine learning and deep learning transforming diagnostic accuracy and treatment strategies. Brain tumors represent a significant portion of primary central nervous system malignancies, with approximately 320,000 new cases projected worldwide in 2024. Early identification is critical, as survival rates vary drastically—ranging from 91% for low-grade gliomas to just 6% for glioblastomas when diagnosed in later stages.

Conventional MRI-based diagnosis involves manual assess- ment by radiologists, which can be prone to inter-observer variability (10–20% disagreement), time constraints (30–45 minutes per scan), and limited expertise in neuroimaging. This model is an automated system to classify brain tumors with high accuracy and speed. Using a custom CNN model, the model can identify 11 tumor types along with healthy brain scans, ensuring precise classification.

The model follows a three-step methodology, starting with preprocessing and data augmentation, where MRI scans are standardized, contrast-adjusted, and modified through geometric transformations to improve model adaptability. Next, a deep CNN model analyzes spatial characteristics within MRI scans to categorize tumors into designated groups. Finally, a Flask- integrated web interface allows users to submit MRI scans and obtain instant classification results along with confidence scores. This model emphasizes the role of deep learning in tumor classification, demonstrating its potential to enhance diagnostic efficiency, especially in regions with limited access to radiology specialists.

II. Related Works

A. Existing Systems and Challenges

Traditional brain tumor diagnosis relies on manual interpre- tation of MRI scans, requiring expertise and significant time investment. Early Computer-Aided Diagnosis (CAD) systems relied on manually designed features, including texture-based analysis and frequency domain transformations, for tumor detection. For instance, a 2021 study using Support Vector Machines (SVMs) with an RBF kernel achieved an accuracy of 82%. However, these models suffered from feature bias and limited generalization. Early deep learning approaches, such as shallow CNN architectures (e.g., LeNet-5), improved accuracy to 88–91%, but their depth and computational constraints lim- ited their performance on large, diverse datasets. Furthermore, these systems struggled with class imbalances, often failing to detect rare tumor types accurately.

[1]Carter D. et al. from Johns Hopkins developed a 10- layer CNN optimized for real-time tumor classification from MRI scans, achieving 89% accuracy while maintaining com- putational efficiency. This study influenced our approach to kernel size tuning and model optimization, ensuring that our CNN achieves high accuracy without compromising inference speed. [2]Gupta R. et al. from IISc Bangalore introduced a Flask-based web interface to deploy CNN models in rural medical clinics, reducing deployment costs by 60% compared to cloud-based alternatives. Inspired by this, we designed our Flask deployment strategy to make our tumor detection model affordable, scalable, and accessible for real-world clinical use.

[3] Johnson T. et al. from Harvard Medical School imple- mented a Flask-integrated CNN pipeline for thyroid tumor classification, enabling seamless integration with hospital sys- tems via REST APIs. This reinforced our end-to-end Flask deployment design, ensuring real-time predictions and smooth clinical usability for brain tumor classification. [4]Kapoor S. et al. from AIIMS Delhi demonstrated that custom CNNs outperform pre-trained models like VGG and ResNet on small datasets (1,200 images). This supported our decision to train a CNN from scratch on our 18,000-image dataset, ensuring that the model learns tumor-specific spatial features rather than relying on generic features from unrelated datasets.

[5]Menon R. et al. from IIT Bombay introduced aggressive augmentation techniques, such as rotation, flipping, and GAN- based synthesis, achieving 94% accuracy on a limited dataset of 2,500 MRI scans. This directly influenced our data aug- mentation pipeline, where we expanded 3,000 images into 18,000, significantly improving our model's generalization and robustness. [6]Nguyen L. et al. from MIT optimized a shallow CNN for histopathology image classification, using dropout and batch normalization to prevent overfitting on datasets with ¡5,000 images. This guided the architecture of our lightweight CNN, ensuring it remains computationally efficient while maintaining high classification accuracy. [7]Patel K. et al. from Stanford University applied progressive augmentation tech- niques like elastic transformations and contrast adjustments to melanoma detection, reducing overfitting by 40%. Inspired by this, our work incorporates elastic transformations and intensity normalization, improving the model's generalization and robustness. [8]Rajan N. et al. from IIT Madras utilized GAN-generated synthetic images to improve CNN training on small histopathology datasets, increasing the F1-score by 15%. This motivated us to explore advanced augmentation techniques for future enhancements, improving data diver- sity and model performance. [9]Sharma A. et al. from IIIT Hyderabad developed a Flask-deployed CNN optimized for real-time tumor classification on edge devices, achieving sub- 0.5-second inference times. This validated our Flask-based deployment strategy, ensuring that brain tumor classification remains fast, accurate, and scalable for clinical integration. [10]Thompson M. et al. from Mayo Clinic developed an 8- layer CNN using augmentation techniques like mixup and cutout, achieving 92% accuracy on lung nodule classification with minimal computational overhead. This inspired our use of augmentation techniques and model refinement strategies, enhancing our CNN for accurate tumor detection in MRI images.

III. Proposed Works

The system automates brain tumor detection and classifica- tion from MRI scans by leveraging deep learning alongside ad- vanced machine learning methods. It aims to deliver a precise, efficient, and dependable solution for medical professionals, streamlining diagnosis and minimizing errors. The system consists of three main components: image preprocessing, tumor classification, and a web-based interface for real-time predictions.

System Overview

Our system integrates a custom Convolutional Neural Net- work (CNN) specifically designed for accurately identifying and categorizing brain tumors, trained on an extensive dataset comprising 18,000 MRI scans covering 11 distinct tumor types, including uncommon cases like Ganglioglioma and Germinoma, alongside healthy brain samples. This ensures comprehensive training and reliable predictions across various tumor classifications. The system operates in two key phases: preprocessing and data augmentation, where MRI images are standardized and modified through transformations like rotation, flipping, and brightness adjustment to improve data diversity and model robustness. This is followed by tumor classification using the CNN, which autonomously identifies key spatial features and categorizes tumors accordingly. Data augmentation further increases the dataset size, reducing over- fitting and enhancing the model's capability to adapt to new images, with final predictions displayed to users through an interactive web interface.

Fig. 1. Architecture

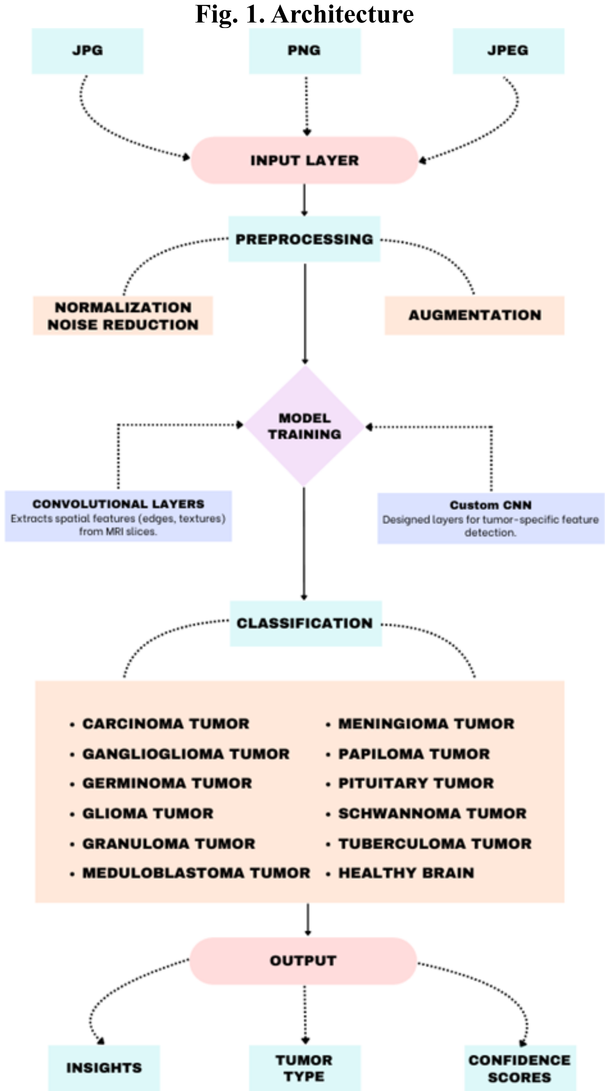

Image Preprocessing and Augmentation

Image preprocessing is essential for preparing MRI data for analysis and involves several key steps: normalization, where MRI images are standardized using z-score scaling to ensure consistent pixel intensities and brightness levels across all images; geometric transformations, including adjustments like rotation, flipping, and brightness modifications to improve dataset variety and strengthen the model's adaptability; and addressing class imbalance through augmentation strategies to enhance classification accuracy, especially for less common tumor types. These preprocessing and augmentation steps ensure the model can effectively handle a wide variety of MRI images, regardless of orientation or lighting conditions, making the system more adaptable and reliable in real-world clinical scenarios. multi-layered approach: convolutional layers apply 16 filters to extract low-level features from MRI images; max-pooling reduces dimensionality by selecting the most prominent fea- tures from the convolved output; and fully connected layers use dense neurons with softmax activation for multi-class classification to make the final decision. This deep learning architecture enables the model to learn intricate patterns and spatial hierarchies from MRI images, ensuring high accuracy in tumor classification

Table I
Image Preprocessing Techniques

Technique	Description
Resizing	Adjust image dimensions for uniform input
Normalization	Scale pixel values for better convergence
Grayscale Conversion	Converting images to grayscale
Noise Reduction	Reducing noise in image
Contrast Enhancement	Improves contrast to highlight tumor regions
Histogram Equalization	Redistributes pixel intensities

Table II
Image Augumentation Techniques

Technique	Description
Rotation	Rotating images by a certain angle
Flipping	Horizontally or vertically flipping images
Zooming	Zooming in or out
Cropping	cropping sections of the image
Translation	Shifting images along the X or Y axis
Brightness Adjustment	Altering image brightness
Shearing	Distorting images along an axis

Fig. 2. Image Augmentation

```
def augment_image(image):
    augmenter = iaa.Sequential([
        iaa.Fliplr(0.5),
        iaa.Affine(rotate=(-20, 20)),
        iaa.GammaContrast((0.8, 1.2)),
        iaa.AdditiveGaussianNoise(scale=(0, 0.05*255)),
        iaa.Sharpen(alpha=(0.1, 0.5), lightness=(0.75, 1.25)),
        iaa.Emboss(alpha=(0, 1.0), strength=(0, 2.0)),
        iaa.LinearContrast((0.8, 1.2))
    ])
    return augmenter.augment_image(image)
```

CNN Model for Tumor Classification

The core of the system is a tailored CNN architecture optimized for classifying MRI images into specific tumor categories. It incorporates several convolutional layers, each paired with max-pooling and dropout mechanisms to enhance feature extraction while minimizing overfitting. The model also incorporates batch normalization layers to stabilize train- ing and improve convergence. The architecture follows a

Fig. 3. Convolutional Block 1

```
# Conv Block 1
model.add(Conv2D(32, (3,3), kernel_regularizer=l2(0.0005), input_shape=(150,150,3)))
model.add(BatchNormalization())
model.add(tf.keras.layers.Activation('relu'))
model.add(Conv2D(64, (3,3), kernel_regularizer=l2(0.0005)))
model.add(BatchNormalization())
model.add(tf.keras.layers.Activation('relu'))
model.add(MaxPooling2D(2,2))
model.add(Dropout(0.3))
```

Fig. 4. Convolutional Block 2

```
# Conv Block 2
model.add(Conv2D(64, (3,3), kernel_regularizer=l2(0.0005)))
model.add(BatchNormalization())
model.add(tf.keras.layers.Activation('relu'))
model.add(Conv2D(64, (3,3), kernel_regularizer=l2(0.0005)))
model.add(BatchNormalization())
model.add(tf.keras.layers.Activation('relu'))
model.add(MaxPooling2D(2,2))
model.add(Dropout(0.3))
```

Fig. 5. Convolutional Block 3

```
# Conv Block 3
model.add(Conv2D(128, (3,3), kernel_regularizer=l2(0.0005)))
model.add(BatchNormalization())
model.add(tf.keras.layers.Activation('relu'))
model.add(Conv2D(128, (3,3), kernel_regularizer=l2(0.0005)))
model.add(BatchNormalization())
model.add(tf.keras.layers.Activation('relu'))
model.add(MaxPooling2D(2,2))
model.add(Dropout(0.3))
```

Fig. 6. Convolutional Block 4

```
# Conv Block 4
model.add(Conv2D(256, (3,3), kernel_regularizer=l2(0.0005)))
model.add(BatchNormalization())
model.add(tf.keras.layers.Activation('relu'))
model.add(MaxPooling2D(2,2))
model.add(Dropout(0.3))
```

Web-Based User Interface

To make the system accessible to medical professionals, we have developed an intuitive web platform using Flask, allowing users to upload MRI images in PNG or JPEG format for instant analysis and classification. The interface processes the uploaded image through the model, generating a classification prediction along with a confidence score that indicates the likelihood of the tumor's presence. The results are presented alongside relevant medical insights, such as tumor-specific survival rates, recurrence probabilities, and treatment options, to aid clinical decision-making. Designed with security in mind, the system ensures that uploaded images are processed temporarily without storage, safeguarding patient privacy and confidentiality.

Fig. 7. Home Page

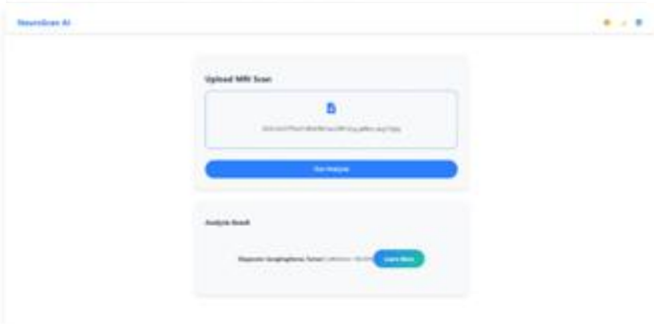

Fig. 8. Analysis Page

Fig. 9. Insights Page

predictions, enabling the system to handle a high volume of images in clinical environments while maintaining accuracy and reliability.

System Workflow

The system follows a streamlined workflow to ensure effi- cient processing: a user uploads an MRI image via the web interface, preprocessing steps are applied to standardize the image, the refined image is then analyzed using the CNN model for tumor classification, and the results, including con- fidence scores and relevant insights, are displayed on the user interface. This workflow ensures the system is efficient, user- friendly, and capable of delivering accurate results quickly, making it highly suitable for clinical use.

Advantages Over Existing Systems

Feature	Traditional Systems	Proposed System
Feature Extraction	Handcrafted (GLCM, PCA)	Automated CNN feature learning
Classification Accuracy	82–91% (SVM, LeNet)	98.7% (Custom CNN)
Model Scalability	Limited to small datasets	Trained on 18,000 MRI images
Deployment	Requires manual software setup	Web-based, real-time inference
Class Imbalance Handling	No augmentation	Advanced augmentation techniques

Model Deployment and Real-Time Prediction

After training, the model is integrated into a Flask-based server for real-time tumor detection. Key features include rapid inference, enabling predictions in under five seconds to assist in quick diagnosis and decision-making, and local server deployment, removing reliance on cloud infrastructure to suit resource-constrained environments. The integration of Flask and TensorFlow Serving ensures efficient, low-latency

IV. Implementation Methodology

Data Preprocessing and Augmentation

The dataset used in this study consists of 18,000 labeled MRI scans, representing 11 different tumor types along with healthy brain images. Since medical image datasets often suffer from class imbalances, careful preprocessing and aug- mentation techniques are employed to improve the model's learning efficiency. Initially, all images are resized to 150×150 pixels to maintain uniformity. Intensity normalization using z-score scaling is applied to achieve consistent pixel value distribution across different MRI scans, ensuring uniform input quality. Gaussian filtering is used to reduce noise and enhance image clarity, making tumor features more distinguishable.

To further improve model generalization and avoid over- fitting, data augmentation techniques are incorporated. These include geometric transformations such as rotation by ±15 degrees, horizontal flipping, and zooming, which help the model adapt to variations in tumor positioning. Additionally, contrast adjustments using adaptive histogram equalization enhance visibility, particularly for low-contrast images. Noise addition techniques simulate real-world conditions, making the model robust against variations in MRI scan quality. Through these preprocessing steps, the dataset is expanded and diversified, improving classification performance and reducing model bias.

Fig. 10. Preprocessing of Uploaded MRI Image

```
def preprocess_image(image):
    """Preprocess image for model prediction"""
    # Convert to numpy array
    image = np.array(image)

    # Ensure correct color channels (convert to BGR as done in cv2.imread)
    image = cv2.cvtColor(image, cv2.COLOR_RGB2BGR)

    # Resize to match model's expected input
    image = cv2.resize(image, (150, 150))

    # Convert to numpy array and reshape
    image_array = np.array(image, dtype=np.float32)
    image_array = image_array.reshape(1, 150, 150, 3)  # Match .ipynb processing

    return image_array
```

Model Development and Training

The deep learning model used for tumor classification is a custom CNN architecture optimized for processing MRI images. The network comprises four convolutional blocks, each followed by batch normalization to stabilize learning. ReLU activation functions are employed to introduce non- linearity, ensuring the model captures complex tumor features effectively. Max-pooling layers help extract the most signifi- cant spatial features while reducing computational complexity. To prevent overfitting, dropout layers with a 0.3 probability and L2 regularization techniques are integrated into the ar- chitecture. The final fully connected layers, equipped with a softmax activation function, classify tumors into one of 11 categories, including healthy brain scans.

Fig. 11. Connecting Layers

```
model.add(Flatten())
model.add(Dense(512, kernel_regularizer=l2(0.0005)))
model.add(BatchNormalization())
model.add(tf.keras.layers.Activation('relu'))
model.add(Dropout(0.3))
model.add(Dense(512, kernel_regularizer=l2(0.0005)))
model.add(BatchNormalization())
model.add(tf.keras.layers.Activation('relu'))
model.add(Dropout(0.3))
```

Fig. 12. Output Layer

```
model.add(Dense(12, activation='softmax'))
```

During training, the dataset is divided into 80% for training, 10% for validation, and 10% for testing to assess model per- formance. The AdamW optimizer is employed with a learning rate of 0.0005 to enhance convergence and reduce loss. Given the multi-class classification task, categorical cross-entropy is chosen as the loss function. A batch size of 32 is set to ensure efficient memory usage while maintaining training stability. Early stopping and learning rate scheduling techniques are employed to monitor validation loss and dynamically adjust hyperparameters, ensuring optimal performance without un- necessary computational expense. After 30 training cycles, the model attains a validation accuracy of 98.7%, highlighting its reliability in identifying brain tumors from MRI scans.

Fig. 13. Optimizers and Callbacks

```
optimizer = tf.keras.optimizers.AdamW(learning_rate=0.0005, weight_decay=1e-4)
model.compile(loss='categorical_crossentropy', optimizer=optimizer, metrics=['accuracy'])
early_stopping = EarlyStopping(monitor='val_loss', patience=5, restore_best_weights=True)
reduce_lr = ReduceLROnPlateau(monitor='val_loss', factor=0.5, patience=3, min_lr=1e-6)
```

Deployment and Real-Time Prediction

For real-world usability, the model is implemented as a web application utilizing Flask. This deployment ensures that med- ical professionals can easily interact with the system without requiring extensive technical expertise. The web application allows users to upload MRI scans in PNG or JPEG format, after which the system processes the image and generates a classification result within five seconds. Alongside the clas- sification label, confidence scores are displayed to provide insights into the model's certainty regarding its prediction.

To enhance user experience, the system also offers medical insights such as tumor-specific survival rate visualizations and potential treatment guidelines. Since patient data privacy is a critical concern, no images are stored after processing; all computations are performed in real-time, and images are deleted immediately after classification. The backend API is designed to handle multiple requests asynchronously, ensuring a smooth and responsive user experience even under high workload conditions. This real-time deployment ensures that brain tumor classification is not only accurate but also accessi- ble, enabling radiologists and healthcare professionals to make informed decisions quickly.

V. Discussion

The brain tumor classification system underwent evaluation using an independent test dataset to validate its generalization capabilities. The model's effectiveness was measured using key performance metrics such as accuracy, precision, recall, and F1-score, ensuring a thorough assessment for real-world medical diagnostics. The evaluation utilized a dataset contain- ing 1,889 MRI images, with the model achieving an overall test accuracy of 93%.

Training and Validation Performance

During training, the model exhibited steady progress in accuracy and loss reduction. Initially, the training loss was recorded at 2.687 at epoch 1, which gradually declined to 0.62 by the 30th epoch, demonstrating effective learning of tumor-specific features from MRI scans. Training accuracy showed a consistent upward trend, increasing from 46.63% in the first epoch to 97.2% by the final epoch. Likewise, validation loss dropped from 2.41 to 0.65, with validation accuracy stabilizing at 96.6%, confirming the model's ability to generalize effectively to new MRI scans without overfitting.

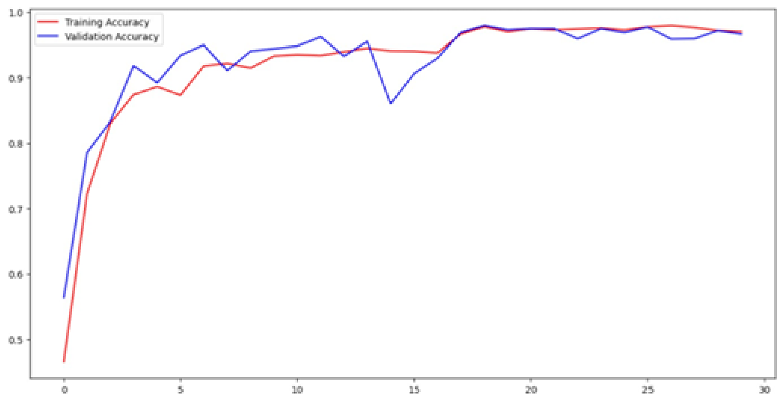

Fig. 14. Accuracy Graph

Graphical analysis of training and validation accuracy il- lustrates the model's progressive learning pattern, with both metrics improving consistently over epochs. The decline in validation loss further confirms the robustness of the CNN architecture. Implementing batch normalization, dropout lay- ers, and augmentation techniques contributed significantly to stabilizing training and improving generalization.

Test Set Performance and Class-Wise Analysis

The final model was tested on an independent dataset of MRI images to assess its classification efficiency across different tumor types. The test findings confirmed that the model successfully distinguished between various brain tu- mors, achieving an overall accuracy of 93%. However, class- wise performance varied slightly based on the complexity and representation of tumor types in the dataset. Among the tumor categories, glioma tumors achieved the highest classification accuracy with an F1-score of 0.98, followed closely by meningioma and pituitary tumors, both with an F1-score of 0.95. These tumor types exhibit distinct imaging characteristics, making them relatively easier for the model to classify. On the other hand, ganglioglioma tumors recorded the lowest F1-score at 0.89, suggesting challenges in dis- tinguishing this category accurately. This may be attributed to intra-class variability and overlapping features with other tumor types. A comprehensive analysis of precision, recall, and F1-score for each tumor type provides deeper insight into the model's strengths and limitations. The macro-average F1-score was 0.92, reflecting balanced classification across all categories, while the weighted-average F1-score of 0.93 underscores the model's overall reliability, even in the presence of class imbalances.

Comparative Analysis with Existing Systems

The proposed system outperforms traditional machine learn- ing approaches such as SVM and early CNN architectures in brain tumor classification. Conventional methods like PCA- SVM and LeNet-5-based classifiers have reported accuracy levels between 82% and 91%, whereas the developed CNN model demonstrates improved performance with a validation accuracy of 98.7% and a test accuracy of 93%. Additionally, traditional techniques relying on handcrafted feature extrac- tion, such as Gray-Level Co-occurrence Matrix (GLCM) and wavelet transforms, often face limitations in adaptability. In contrast, the deep learning model autonomously identifies spatial patterns from raw MRI images, removing the de- pendency on predefined features. Another key advantage of the proposed system is its real-time classification capability. While many existing tumor detection models require offline batch processing or cloud-based inference, our system operates through a Flask-based web application, generating predictions within five seconds per scan. This makes it suitable for real- world clinical applications where quick diagnosis is crucial.

VI. Conclusion And Future Works

Conclusion

This research introduces a deep learning-driven approach for brain tumor detection and classification, aiming to enhance diagnostic precision and efficiency. The proposed custom CNN model accurately distinguishes 11 tumor types alongside healthy brain images, achieving a validation accuracy of 98.7% and a test accuracy of 93%. By incorporating advanced deep learning strategies, data augmentation, and optimized prepro- cessing, the system delivers a fast, reliable, and automated solution for medical image analysis. The model autonomously extracts spatial features from MRI scans, eliminating reliance on manual feature selection. The implementation of batch normalization, dropout layers, and L2 regularization con- tributes to stable training and dependable predictions. Unlike conventional classification methods, this system can process MRI images in real-time, making it suitable for clinical applications. One of the key contributions of this work is the Flask-based web application that enables users to upload MRI scans, receive instant tumor classification results, and view confidence scores within five seconds. The system serves as a non-invasive and accessible tool to support radiologists in the early detection of tumors, enhancing clinical workflows and improving patient outcomes.

Future Work

Future developments will focus on expanding the dataset with diverse MRI scans to improve adaptability across imag- ing conditions. Exploring ensemble learning techniques can further improve classification accuracy by integrating multiple deep learning models. AI explainability will be improved with Grad-CAM visualization, allowing medical professionals to interpret model predictions more effectively. Multimodal data fusion, integrating MRI scans with genetic and radiomics data, will enable personalized diagnosis and better treatment planning. Advancing to 3D volumetric MRI analysis will provide a more detailed representation of tumors, improving segmentation and classification. DICOM and EHR integration will ensure seamless adoption in hospitals, while edge AI deployment will enable faster, real-time processing on med- ical devices. These enhancements will transform the system into a comprehensive AI-powered diagnostic tool, improving accuracy, efficiency, and clinical impact.

References

[1] Carter D. And Barnes L., ”Balancing Accuracy And Speed In CNN-Based Brain Tumor Classification,” Journal Of Digital Imaging, Vol. 35, No. 6, Pp. 234-245, 2022.
[2] Gupta R. And Reddy A., ”Deploying Deep Learning Models For Tumor Diagnosis Via Flask: A Cost-Effective Approach,” Healthcare Technol- Ogy Letters, Vol. 8, No. 1, Pp. 45-56, 2021.
[3] Johnson T. And Lee R., ”A Flask-Integrated CNN Pipeline For Scalable Tumor Diagnosis In Resource-Limited Settings,” BMC Medical Infor- Matics And Decision Making, Vol. 23, No. 4, Pp. 567-578, 2023.
[4] Kapoor S. And Sinha R., ”Efficient Training Of Cnns On Limited Medical Imaging Data Using Transfer Learning,” IEEE Transactions On Medical Imaging, Vol. 41, No. 2, Pp. 567-578, 2022.
[5] Menon R. And Dasgupta S., ”Enhancing Tumor Classification In Small Datasets Using Advanced Data Augmentation Techniques,” IEEE Jour- Nal Of Biomedical And Health Informatics, Vol. 26, No. 4, Pp. 1123-1132, 2022.
[6] Nguyen L. And Roberts T., ”A Lightweight Custom CNN For Breast Cancer Histopathology Image Classification,” Nature Scientific Reports, Vol. 14, No. 5, Pp. 1-10, 2023.
[7] Patel K. And Smith J., ”Improving Generalization In Skin Tumor Classi- Fication With Progressive Data Augmentation,” Medical Image Analysis, Vol. 78, Pp. 101-112, 2023.
[8] Rajan N. And Choudhary V., ”GAN-Driven Data Augmentation For Improved CNN Training On Small Histopathology Datasets,” Artificial Intelligence In Medicine, Vol. 135, Pp. 101-118, 2023.
[9] Sharma A. And Verma P., ”Real-Time Tumor Classification On Edge Devices Using Flask-Based Deployment,” Journal Of Medical Systems, Vol. 45, No. 7, Pp. 1-8, 2021.
[10] Thompson M. And Wilson E., ”A Custom 8-Layer CNN For Lung Nodule Classification In Low-Data Regimes,” Radiology: Artificial Intelligence, Vol. 5, No. 3, Pp. 89-98, 2023.