http://doi.org/10.35784/iapgos.6035 received: 24.03.2024 | revised: 24.05.2024 | accepted: 04.09.2024 | available online: 30.09.2024

MODIFIED VGG16 FOR ACCURATE BRAIN TUMOR DETECTION IN MRI IMAGERY

Katuri Rama Krishna, Mohammad Arbaaz, Surya Naga Chandra Dhanekula, Yagna Mithra Vallabhaneni
Velagapudi Ramakrishna Siddhartha Engineering College, Department of Computer Science and Engineering, Vijayawada, India

Abstract. *Brain tumors are one of the most severe medical conditions that require immediate attention and treatment. The early detection of brain tumors is of utmost importance, as it can significantly improve the chances of successful treatment outcomes and increase the patient's quality of life. This study proposes a novel methodology for the early detection of brain tumors in magnetic resonance imaging (MRI) images using a modified VGG16 neural network architecture. The dataset comprises both tumor and non-tumor MRI images collected from Kaggle and has preprocessing techniques applied to optimize the model's performance. The proposed approach delivers an impressive accuracy rate of 99.08%, demonstrating its efficacy in precise brain tumor detection. This new methodology is expected to aid doctors in accurate diagnosis and treatment planning, thereby helping to save more lives and improve the quality of life of patients suffering from brain tumors.*

Keywords: brain tumor, deep learning, VGG16, detection

ZMODYFIKOWANY VGG16 DO DOKŁADNEGO WYKRYWANIA GUZÓW MÓZGU W OBRAZACH MRI

Streszczenie. *Guzy mózgu są jednym z najpoważniejszych schorzeń, które wymagają natychmiastowej uwagi i leczenia. Wczesne wykrywanie guzów mózgu ma ogromne znaczenie, ponieważ może znacznie zwiększyć szanse na pomyślne wyniki leczenia i poprawić jakość życia pacjenta. W niniejszym badaniu zaproponowano nowatorską metodologię wczesnego wykrywania guzów mózgu na obrazach rezonansu magnetycznego (MRI) przy użyciu zmodyfikowanej architektury sieci neuronowej VGG16. Zbiór danych obejmuje zarówno obrazy MRI guza, jak i nienowotworowe zebrane z Kaggle i zawiera techniki wstępnego przetwarzania zastosowane w celu optymalizacji wydajności modelu. Zaproponowane podejście zapewnia imponującą dokładność na poziomie 99,08%, wykazując swoją skuteczność w precyzyjnym wykrywaniu guzów mózgu. Oczekuje się, że ta nowa metodologia pomoże lekarzom w dokładnej diagnozie i planowaniu leczenia, pomagając w ten sposób uratować więcej istnień ludzkich i poprawić jakość życia pacjentów cierpiących na guzy mózgu*

Słowa kluczowe: guz mózgu, uczenie głębokie, VGG16, wykrywanie

Introduction

The human body is a complex system comprised of several cell types that collaborate harmoniously to maintain its health and proper functioning. Every individual cell possesses a distinct purpose and undergoes a systematic process of growth and division to generate new cells. Occasionally, certain cells have a loss of regulatory control over their proliferation, resulting in uncontrolled development and the formation of a neoplasm referred to as a tumor. Tumors can exhibit either malignant or benign characteristics. Benign tumors are non-malignant and do not metastasize, whereas malignant tumors can metastasize, resulting in the development of cancer. According to the findings of the Central Brain Tumor Registry of the United States (CBTRUS), an estimated 39,550 individuals were diagnosed with either benign or malignant brain tumors in the year 2002. This underscores the significance of comprehending the etiology and consequences of aberrant cellular proliferation, as well as the imperative for timely identification and intervention [8].

Magnetic Resonance Imaging (MRI) is a commonly used diagnostic tool for brain tumors [5]. MRI images can provide valuable information about the tumor's location, size, and shape, allowing doctors to make informed decisions about treatment options. However, MRI images have limitations in detecting certain types of brain tumors. In such cases, advanced machine learning techniques, such as convolutional neural networks (CNNs), can be used in conjunction with these images to improve the accuracy and reliability of the diagnosis. CNNs have proven highly effective and accurate in detecting, classifying, and segmenting brain tumors [4]. With the ability to analyse large amounts of medical image data, CNN can detect subtle patterns and identify abnormalities that may go unnoticed by human experts. In addition, CNN is an invaluable tool for the diagnosis and treatment of brain cancers since it may draw on past experiences to enhance its precision and effectiveness. The pre-trained Visual Geometry Group (VGG16) CNN model is enhanced with customized layers in this study's proposed approach for accurately detecting brain cancers.

1. Literature review

Mohsen et al. [7] have recently developed, utilizing the Ten-sorFlow library, a cutting-edge model for autonomously detecting brain tumors. The model incorporates Conv2D layers with intentional kernel sizes and output shapes, in addition to CNNs. Furthermore, dropout and maxpooling2D layers are incorporated to guarantee regularization. They enhanced the efficacy of the model by fine-tuning several hyperparameters. The CNN was trained employing the Python programming language in Google Colab on a high-speed laptop equipped with an Intel Core i3 processor. To achieve optimal results, they additionally delineated the training iterations, sample size, and total trainable parameters. Furthermore, the dataset utilized for both training and testing purposes was partitioned 80% by 20%. The model has been developed to distinguish between images of pituitary tumors and meningiomas by analysing pixel intensity and feature information. Their method has demonstrated encouraging outcomes in the automated detection of brain tumors, achieving an accuracy rate of 95.78%. This presents an opportunity for early diagnosis and treatment.

In a recent study, Swaroop et al. [9] presented a novel approach for detecting brain tumors using an ensemble learning technique that combines CNN, AlexNet, and GoogLeNet. The study aimed to develop a pipeline that leverages deep learning models to detect tumors in the brain with high accuracy and low cost. The researchers used four deep learning models to achieve this goal and analyzed various system parameters to determine the optimal recognition system. They also employed data augmentation techniques to overcome the challenges of data scarcity and imbalance, which resulted in a more extensive dataset for training neural network models. The proposed framework focused on improving image segmentation and spatial confinement, leading to better accuracy and efficiency of brain tumor detection through advanced convolutional neural network architectures. The study reported an accuracy of 76.37% for the CNN model, which is a promising result for future research in this field.

Ayomide et al. [1] proposed an enhanced methodology for brain tumor segmentation in MRI images using CNNs and MATLAB's GoogLeNet, incorporating anisotropic diffusion filtering, morphological operations, and sector vector machine. The modified GoogLeNet with 22 layers was trained for precise tumor segmentation, achieving high accuracy. Additionally, a CNN-based U-net with transfer learning from VGG16 was employed for tumor segmentation and grading, enhancing the classification model. The SVM classifier with kernel functions was utilized for image segmentation and data transformation. The methodology also included anisotropic diffusion for noise reduction and iterative processes for final image output. The hybrid CNN-SVM approach demonstrated superior performance in tumor classification, showcasing high accuracy and effectiveness.

A methodology was put forth by Younis et al. [10] to automate the identification of brain tumors in MRI scans through the utilization of CNN, VGG 16, and an ensemble model. The research concentrated on attaining high levels of accuracy; CNN achieved 96% accuracy, VGG 16 achieved 98.15% accuracy, and the Ensemble Model achieved 98.41% accuracy. The researchers applied optimization techniques and targeted hyperparameters to a training dataset consisting of 80% for training, 10% for validation, and 10% for testing to improve the performance of the models. Accuracy, recall, and F1-score were utilized as evaluation metrics to demonstrate that the proposed method accurately detects brain lesions in MRI images.

Pillai et al. [6] have proposed a novel methodology utilizing deep transfer learning models to detect brain tumors. Three distinct models were employed for the purpose of enhancing accuracy: VGG16, InceptionV3, and ResNet50. To refine these models, Dropout, Flatten, and Dense layers were incorporated. The models were trained using a total of 251 MRI scans. VGG16 demonstrated superior performance compared to the alternative models, achieving an accuracy rate of 91.58%. The results of the study indicate that deep learning models have the capability to accurately identify brain lesions while requiring minimal resources and time.

A comprehensive study was undertaken by Gayathri et al. [3] in an effort to develop a practical method for identifying brain tumors with precision. Utilizing the VGG16 architecture and a dataset comprised of 1598 scans devoid of tumors and 1655 scans containing tumors, the researchers obtained the data from well-known repositories such as TCIA and Kaggle. In order to preprocess the images, they were resized to 224x224 pixels and normalized. Through transfer learning, the pre-trained VGG16 model was refined, allowing it to execute the task with greater efficiency. The efficiency of the model was assessed by the researchers through the utilization of various metrics – sensitivity, specificity, precision, recall, and F1 Score – which collectively indicated that the algorithm exhibited exceptional accuracy in the detection of brain tumors. The research emphasized the advantages of employing the VGG16 framework for effective image recognition endeavors and proposed various approaches to improve the model's applicability beyond its immediate domain and its clinical significance in the detection of brain tumors.

2. Proposed methodology

The methodology consisted of three sections: Dataset collection, Data pre-processing, and Modifications to the VGG16 model, as shown in Fig. 1.

2.1. Dataset collection

The study utilized MRI data sourced from Kaggle [2]. The dataset was well-organized into distinct folders, facilitating easy categorization into separate training and testing data. Each of these folders also contained subfolders that represented specific tumor classes. However, before uploading the dataset to the Colab environment, a preprocessing step was carried out to ensure uniformity and balance within the dataset. This step involved removing excess images and standardizing the number of MRI images in each class folder. As a result, the 'no' folder contained 396 MRI images, while the 'yes' folder also contained an equal count of 396 images. Fig. 2 shows a sample of what the dataset looks like. This careful selection and curation of the dataset aimed to create a balanced and representative training process for machine learning model development.

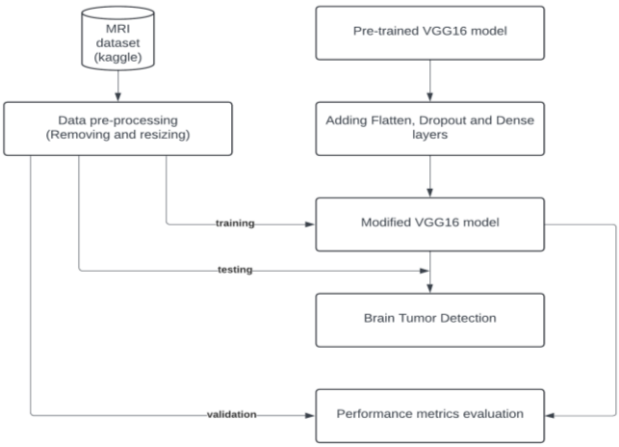

Fig. 1. Methodology for detecting brain tumor in MRI images

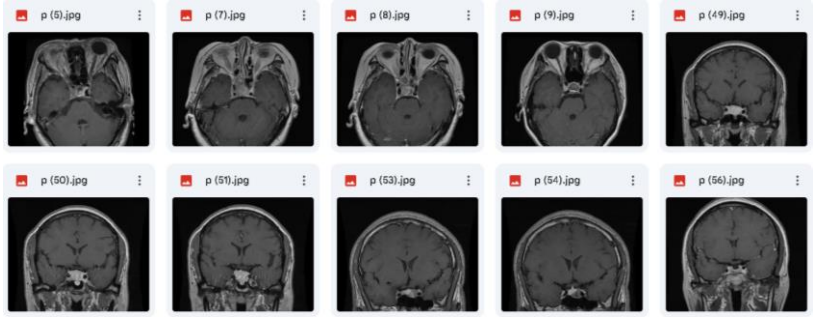

Fig. 2. Samples of the Kaggle dataset

2.2. Data pre-processing

During the data preprocessing phase of the brain tumor detection model's development, several essential steps were taken to ensure that the dataset was adequately prepared for training and evaluation. Firstly, we structured the dataset into distinct directories for training, testing, and validation, strategically placed along specific paths within the Colab environment. This organization facilitated easy and efficient access to the data during the development process. We resized the image data to a standard dimension of 224x224 pixels using the *ImageDataGenerator* from TensorFlow's Keras library to ensure uniformity in image dimensions and practical VGG16 model training.

We rescaled the pixel values of the images to a range between 0 and 1 to normalize the data and aid in model convergence during training. To prevent class imbalances that could affect model performance, we stratified the dataset by distributing images across the train, test, and val sets according to 70%, 20%, and 10% percentages. This process ensured that each set maintained a proportional representation of the different classes ('yes' and 'no' in this case) in the dataset.

Finally, we moved the images from their source folder to their respective destination folders within the training, testing, and validation directories to ensure compatibility with the subsequent steps in the machine learning pipeline. This relocation process organized the dataset into the appropriate structure required by the *flow_from_directory* function. We deemed the dataset ready for model training and evaluation after completing these preprocessing steps.

As part of the analysis of the generated datasets, we conducted an in-depth assessment of class distribution across multiple dataset subsets, including the training, testing, and validation sets. We mapped the class indices to their corresponding class names by inverting dictionaries, which enabled a comprehensive understanding of the dataset's composition. We computed the sample sizes for each class to determine the equilibrium and representation of the dataset across various categories.

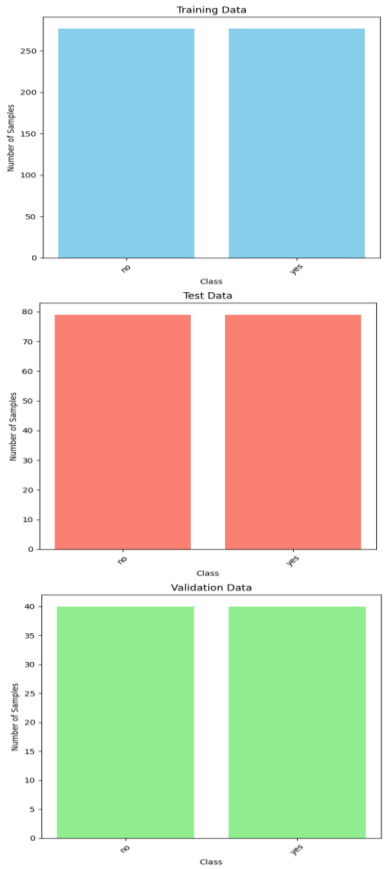

Fig. 3. Class distribution of train, test and val sets

Fig. 3 uses bar plots to visually depict the class distribution, specifically highlighting the sample distribution across classes for each subset. The analysis revealed that the dataset exhibited a balanced composition, as each class had an equivalent number of samples. This analysis offers significant insights into the attributes of the dataset, thereby facilitating the interpretation of the model's efficacy and capacity for generalization.

2.3. Modifications to the VGG16 model

We used the VGG16 architecture as a base to build and assemble a CNN model for binary classification tasks. The first step was to load the pre-trained VGG16 model, already trained on the ImageNet dataset, without its fully connected layers. This enabled us to utilize the model's learned feature extraction capabilities. We froze the weights of all layers to preserve the integrity of the learned features and avoid updating the pre-trained weights during training.

Next, we designed a new model architecture with additional layers specifically tailored for the classification task. This architecture included a flattening layer, followed by dropout regularization, which helped to mitigate overfitting. The model also incorporated a dense layer with ReLU activation, specifically designed for feature extraction. We also introduced an extra dropout layer to further enhance the model's generalization. Finally, we appended a dense output layer with sigmoid activation to facilitate binary classification.

The model summary, as shown in Fig. 4, provides in-depth insights into the network's architecture, detailing the structure of each layer and the number of trainable parameters used. After constructing the model, the compilation phase involved specifying the optimizer, loss function, and evaluation metric. The Adam optimizer and binary cross-entropy loss are chosen to optimize and assess the model's performance.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 vgg16 (Functional)          (None, 7, 7, 512)         14714688

 flatten (Flatten)           (None, 25088)             0

 dropout (Dropout)           (None, 25088)             0

 dense (Dense)               (None, 256)               6422784

 dropout_1 (Dropout)         (None, 256)               0

 dense_1 (Dense)             (None, 2)                 514

=================================================================
Total params: 21137986 (80.64 MB)
Trainable params: 6423298 (24.50 MB)
Non-trainable params: 14714688 (56.13 MB)
_________________________________________________________________
```

Fig. 4. Modified VGG16 model's summary

We set the number of epochs and batch size to 10 and 32, respectively, to initiate the model's training. We then invoked the *fit* method on the model, using the *train_generator* as the input data source. The model underwent iterative optimization during training across multiple epochs, each representing a complete pass through the entire training dataset. We chose the batch size to determine the number of samples processed in each training iteration, thereby facilitating efficient model optimization. We also provided validation data from the *val_generator* to prevent the model from overfitting. This allowed for assessing the model's performance on unseen data during training and the ability to monitor the model's generalization capabilities. The goal was to iteratively adjust the model parameters and minimize the specified binary cross-entropy loss function. Ultimately, the aim was to optimize the model for accurate brain tumor detection.

3. Results and analysis

As shown in Fig. 5, a line plot visualized the training history, showing the evolution of loss values throughout training epochs. The plot showcased the training loss and validation loss on the y-axis against the number of epochs on the x-axis. The training loss curve illustrated the gradual decrease in loss as the model iteratively optimized its parameters over successive epochs. Concurrently, the validation loss curve depicted the model's performance on unseen validation data, offering insights into its generalization capabilities. The lack of significant divergence between the training and validation loss curves indicated practical model training without substantial overfitting. This visualization facilitated monitoring model convergence and generalization during the training process, guiding further model refinement and optimization efforts.

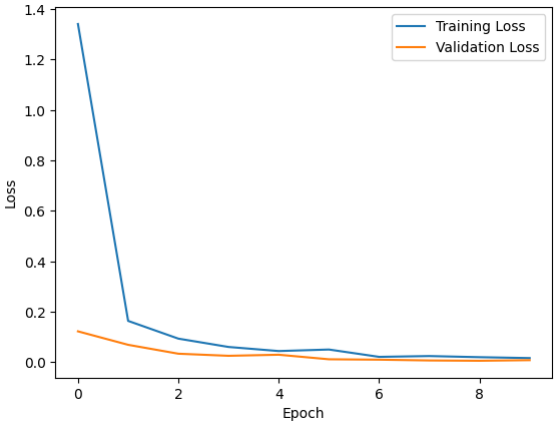

Fig. 5. Training and Validation loss plot of the model

Fig. 6 illustrates the generation of the training history diagram for accuracy. Its purpose was to monitor the performance of the model throughout the training and validation epochs, analogous to the loss visualization. The y-axis of the graph represented the training and validation accuracy, while the x-axis represented the number of epochs. The training accuracy curve exhibited a progressive ascent over the course of the training process, signifying the model's advancing capability to accurately classify samples from the training dataset. In a similar vein, the validation accuracy curve revealed the model's capacity to generalize across successive epochs by revealing its performance on unobserved validation data. With this visualization, it was easier to see how model convergence and generalization were changing over time. Both of these things are needed to make the model work better and find problems like overfitting.

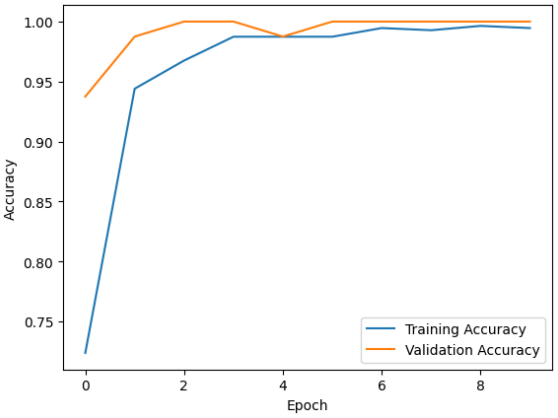

Fig. 6. Training and Validation accuracy plot of the model

Utilizing the seaborn and matplotlib libraries, a heatmap visualization was produced to showcase the model's classification performance through the confusion matrix. We plotted the confusion matrix (Fig. 7) heatmap with 'yes' and 'no' class names on both axes, representing true positive, true negative, false positive, and false negative classifications. Annotations included within the heatmap aided in interpreting the frequency of each classification outcome, while the color gradient allowed for easy identification of classification patterns. This visualization technique provided significant insights into the model's capacity to precisely classify brain tumor images while highlighting areas for improving classification accuracy.

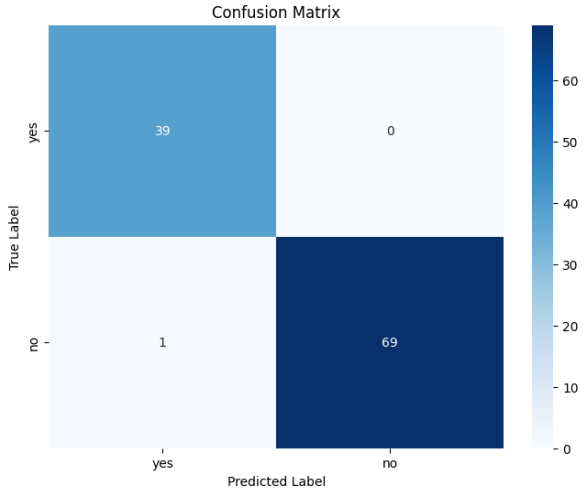

Fig. 7. Confusion matrix heatmap of the model

Table 1 describes the model's accuracy, precision, recall, and f1-score values.

Table 1. Performance metrics of the modified VGG16 model

Model	Accuracy (%)	Precision	Recall	F1-score
Modified VGG16	99.08	1.00	0.98	0.99

We loaded and pre-processed a sample image from the validation dataset to ensure its compatibility with the model's input requirements. We then fed the pre-processed image into the trained model, which used a binary classification assumption with a threshold of 0.5 to predict its class label. Finally, the sample image was displayed alongside its predicted class label ('no'), visually representing the model's classification output for the given image, as shown in Fig. 8.

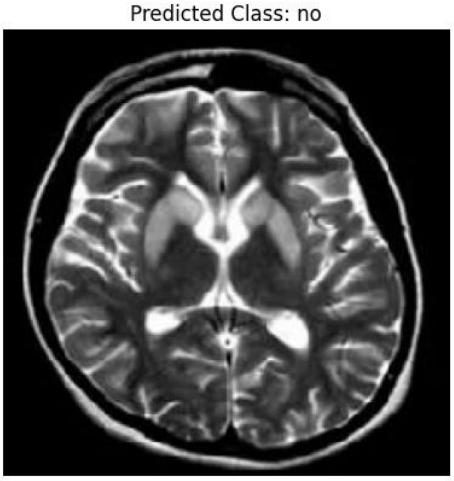

Fig. 8. Classification output of the model

4. Conclusion

The results of using a modified VGG16 neural network architecture for detecting brain tumors were promising, as demonstrated by an impressive 99.08% accuracy. The precision metric, which measures the model's ability to identify positive cases correctly, achieved a perfect score of 1.00, indicating minimal false positive classifications. The model also displayed a strong recall performance of 0.98, demonstrating its ability to accurately capture the most favourable cases. The computed F1-score, a harmonic mean of precision and recall, further emphasized the model's robust performance, reaching an exceptional value of 0.99. These findings underline the effectiveness of the proposed approach in accurately detecting brain tumors from MRI images, showcasing its potential as a valuable tool in clinical settings. The model's superior performance metrics highlight its reliability and effectiveness, underscoring its potential to assist medical professionals in timely and accurate diagnosis, ultimately enhancing patient care and treatment outcomes.

References

[1] Ayomide K. S. et al.: Improving Brain Tumor Segmentation in MRI Images Through Enhanced Convolutional Neural Networks. International Journal of Advanced Computer Science and Applications 14(4), 2023.
[2] Brain Tumor Classification (MRI). Kaggle (24 May 2020), [www.kaggle.com/datasets/sartajbhuvaji/brain-tumor-classification-mri].
[3] Gayathri P. et al.: Exploring the Potential of VGG16 Architecture for Accurate Brain Tumor Detection Using Deep Learning. Journal of Computers, Mechanical and Management 2(2), 2023.
[4] Hemanth G. et al.: Design and Implementing Brain Tumor Detection Using Machine Learning Approach. 3rd International Conference on Trends in Electronics and Informatics –cICOEI. IEEE, 2019.
[5] Kapoor L., Sanjeev T.: A Survey on Brain Tumor Detection Using Image Processing Techniques. 7th International Conference on Cloud Computing, Data Science & Engineering – Confluence. IEEE, 2017.
[6] Pillai R. et al.: Brain Tumor Classification Using VGG 16, ResNet50, and Inception V3 Transfer Learning Models. 2nd International Conference for Innovation in Technology – INOCON. IEEE, 2023.
[7] Saeed M. et al.: A Convolutional Neural Network for Automatic Brain Tumor Detection. Engineering and Technology Innovation 24, 2023, 15–21.
[8] Sharma K. et al.: Brain Tumor Detection Based on Machine Learning Algorithms. International Journal of Computer Applications 103(1), 2014, 7–11.
[9] Swarup C. et al.: Brain Tumor Detection Using CNN, AlexNet and GoogLeNet Ensembling Learning Approaches. Electronic Research Archive 31(5), 2023, 2900–2924.
[10] Younis A. et al.: Brain Tumor Analysis Using Deep Learning and VGG16 Ensembling Learning Approaches. Applied Sciences 12(14), 2022, 7282.

Prof. Katuri Rama Krishna
e-mail: kramakrishna@vrsiddhartha.ac.in

Professor K. Rama Krishna is currently employed as an assistant professor in the Department of Computer Science and Engineering at V. R. Siddhartha Engineering College in Vijayawada. He is pursuing a Ph.D. in Computer Science and Engineering from Jawaharlal Nehru Technological University, Kakinada. His research interests include Image Processing, Data Mining, and Machine Learning.

https://orcid.org/0000-0002-1970-736X

Eng. Mohammad Arbaaz
e-mail: mohdarbaaz300092@gmail.com

Mohammad Arbaaz is a final-year student pursuing a B. Tech in Computer Science and Engineering at V. R. Siddhartha Engineering College, Vijayawada. He has a keen interest in Machine Learning.

https://orcid.org/0009-0007-3170-2039

Eng. Surya Naga Chandra Dhanekula
e-mail: snchandradhanekula@gmail.com

Surya Naga Chandra Dhanekula is a final year B. Tech student specializing in Computer Science and Engineering at V. R. Siddhartha Engineering College, Vijayawada. He is interested in projects related to Deep learning.

https://orcid.org/0009-0009-2790-1451

Eng. Yagna Mithra Vallabhaneni
e-mail: vymithra@gmail.com

Yagna Mithra Vallabhaneni is in his final year of Computer Science and Engineering at V. R. Siddhartha Engineering College, Vijayawada. His academic specialization focuses on Machine Learning and Deep Learning projects.

https://orcid.org/0009-0005-0842-315X