Jurnal Teknik Informatika Vol. 18 No. 1, April 2025 (87-100)
ISSN: p-ISSN 1979-9160 (Print)| e-ISSN 2549-7901 (Online)
DOI: https://doi.org/10.15408/jti.v18i1.44147

JURNAL TEKNIK INFORMATIKA

Homepage : http://journal.uinjkt.ac.id/index.php/ti

Impact of Hyperparameter Tuning on CNN-Based Algorithm for MRI Brain Tumor Classification

Muhammad Nasri Gea[1*], Wanayumini[2], Rika Rosnelly[3]

[1,3]Computer Science, Computer Science and Engineering, Potensi Utama University
[2]Computer Science, Computer Science and Engineering, Asahan University
[1,3]Jl. K.L. Yos Sudarso KM 6,5 No. 3A Tj. Mulia, Indonesia
[2]Jl. Jend.A.Yani, Kisaran Naga, Kec. Kota Kisaran Timur, Kisaran, Sumatera Utara, Indonesia

Article:
Accepted: January 18, 2025
Revised: October 12, 2024
Issued: April 30, 2025

***Correspondence Address:**
muhammadnasrig@gmail.com

ABSTRACT

This study examines the impact of hyperparameter tuning on the performance of Convolutional Neural Networks (CNN) in classifying brain tumors using MRI images. The dataset, sourced from Kaggle, underwent preprocessing techniques such as normalization, augmentation, and resizing to enhance consistency and diversity. The study evaluates five hyperparameter configurations, analyzing their effects on classification accuracy, precision, recall, and F1-score. The optimal configuration (batch size: 16, epochs: 10, learning rate: 0.001) achieved an accuracy of 86%, precision of 81%, recall of 85%, and an F1-score of 0.83. Other configurations showed trade-offs, where larger batch sizes increased recall but reduced precision. These findings emphasize the importance of careful hyperparameter tuning to optimize medical imaging classification performance.

Keywords : *hyperparameter tuning; convolutional neural networks; brain tumor detection; MRI image classification; machine learning; deep learning.*

Jurnal Teknik Informatika Vol. 18 No. 1, April 2025 (87-100)
ISSN: p-ISSN 1979-9160 (Print)| e-ISSN 2549-7901 (Online)
DOI: https://doi.org/10.15408/jti.v18i1.44147

1. INTRODUCTION

Deep learning, a subfield of artificial intelligence (AI), has revolutionized various domains, particularly in medical imaging. Convolutional Neural Networks (CNNs) have been widely adopted for image classification tasks due to their ability to automatically extract spatial features from images without requiring manual feature engineering [1]. In medical applications, CNNs have shown promising results in tasks such as tumor detection, disease classification, and anomaly identification [2].

Hyperparameter tuning plays a crucial role in optimizing CNN performance, affecting factors such as accuracy, precision, recall, and computational efficiency. Key hyperparameters in CNN models include learning rate, batch size, number of epochs, kernel size, and the number of filters [3]. Improper selection of these hyperparameters can lead to issues like overfitting, slow convergence, or suboptimal classification performance [4]. In CNN architectures, the kernel size determines the spatial area for feature extraction, where an overly small kernel may lose critical spatial information, and an excessively large kernel can increase computational complexity and lead to overfitting [5]. Similarly, the number of filters directly impacts the depth of learned features, affecting the model's ability to generalize across different tumor patterns [6]. The learning rate is another crucial hyperparameter that controls how quickly the model updates weights during training; an excessively high value may prevent convergence, whereas a very low value can lead to prolonged training and suboptimal results [7].

Previous studies have demonstrated CNN's effectiveness across various domains, including satellite image analysis [8], autonomous vehicle object recognition [9], and disease detection in medical imaging [10]. However, despite these advancements, limited research has systematically analyzed the role of hyperparameter tuning in CNN-based MRI brain tumor classification.

This study aims to address this gap by evaluating the effect of different hyperparameter configurations on CNN performance in detecting brain tumors from MRI scans. By systematically analyzing hyperparameters such as batch size, learning rate, and epochs, this research seeks to identify an optimal configuration that balances precision and recall while maintaining computational efficiency.

Several previous studies have explored hyperparameter tuning in CNN-based classification tasks. For example, [11] optimized CNN hyperparameters for meat type classification, achieving an accuracy of 82.06% on 75×75 px images. In cassava disease classification, [12] found that the Adam optimizer outperformed SGD in terms of convergence speed and accuracy. Similarly, [13] applied random search for CNN hyperparameter optimization in COVID-19 detection from chest X-ray images, achieving an accuracy range of 88%-95.38% while reducing computational cost. However, these studies did not specifically address MRI-based tumor detection.

More relevant studies, such as [14] and [15], investigated optimal epoch counts and batch sizes for wildfire and paddy disease classification, respectively. While these works highlight the importance of hyperparameter tuning, they do not directly address the unique challenges of medical image classification, particularly MRI scans, which require specialized preprocessing techniques and higher sensitivity to false negatives.

To bridge this gap, this research systematically evaluates the impact of hyperparameter tuning on MRI brain tumor classification using CNNs. The findings aim to improve classification accuracy, enhance model generalization, and contribute to advancements in automated medical diagnostics.

2. METHODS

Here is the research workflow for "Impact of Hyperparameter Tuning on CNN-Based Algorithm for MRI Brain Tumor Classification" along with a diagram illustrating the steps :

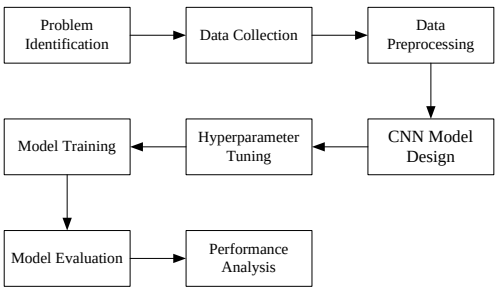

Figure 1. *Research framework flowchart*

Figure 1 presents the Research Framework Flowchart, illustrating the sequential steps undertaken in this study. Each stage is interconnected, ensuring a systematic approach to analyzing hyperparameters for CNN-based MRI brain tumor classification. Descriptions of each process are as follows:

a. Problem Identification

The research begins with identifying the core challenge : accurately classifying MRI brain images into tumor and non-tumor categories. Early and precise detection is crucial for effective medical diagnosis. This study aims to optimize CNN hyperparameters to enhance model accuracy and reliability, thereby improving automated tumor detection in medical imaging.

b. Data Collection

The dataset used in this study is sourced from Kaggle (Navoneel/brain-mri-images-for-brain-tumor-detection). Kaggle is a leading platform that allows access to various types of datasets for research, machine learning, and other data science projects [16]. Taking datasets from Kaggle is a common practice in data science research because Kaggle provides a variety of high-quality datasets that can be used for various research purposes [17]. It contains MRI scans with and without brain tumors, covering various tumor types, sizes, and locations. The dataset is balanced to prevent classification bias and ensure robust model performance.

c. Data Preprocessing

Data preprocessing is a critical step in the data science pipeline that can significantly impact the final outcome of the analysis [18]. The preprocessing process is carried out to clean data from noise, reduce data dimensions, and make data more structured [19]. Data preprocessing is a step to ensure the MRI images are suitable for CNN training. The following preprocessing techniques are applied :

1. Normalization
   Pixel intensity values are scaled to a range of 0-1 for uniformity.
2. Data Augmentation
   Techniques such as rotation, flipping, and scaling are applied to enhance data diversity and mitigate overfitting.
3. Noise Removal
   Image filtering is employed to remove noise and artifacts.
4. Resizing
   All images are resized to a fixed resolution suitable for CNN input layers, ensuring consistent feature extraction.

d. CNN Model Design

A CNN architecture is developed to classify MRI brain images. The main components in CNN include : Convolution Layer, Pooling Layer, Fully Connected Layer, and Activation Function [20]. The model consists of the following key components :

1. Convolutional Layers
   Extract spatial features from MRI images.
2. Pooling Layers
   Reduce dimensionality and retain essential features.
3. Fully Connected Layers
   Convert extracted features into classification decisions.
4. Activation Functions
   ReLU is applied in convolutional layers, while Softmax is used in the output layer for tumor classification.

This study explores different CNN architectures by varying hyperparameters such as kernel size, the number of filters, and dropout rates to optimize model performance.

e. Hyperparameter Tuning

HyperParameter Tuning is a process in machine learning that optimizes model performance by systematically testing different ons of parameters to determine which ones produce the best results on a given dataset [21]. This process involves adjusting model parameters such as kernel size, threshold value, and others to maximize model performance [22]. Hyperparameter optimization is performed using grid search, systematically testing various configurations to determine the best-performing combination. The following hyperparameter settings are evaluated :

1. Batch Size : 16, Epochs : 10, Learning Rate : 0.001
2. Batch Size : 32, Epochs : 10, Learning Rate : 0.001
3. Batch Size : 16, Epochs : 15, Learning Rate : 0.0001
4. Batch Size : 32, Epochs : 15, Learning Rate : 0.0001

Jurnal Teknik Informatika Vol. 18 No. 1, April 2025 (87-100)
ISSN: p-ISSN 1979-9160 (Print)| e-ISSN 2549-7901 (Online)
DOI: https://doi.org/10.15408/jti.v18i1.44147

5. Batch Size : 64, Epochs : 15, Learning Rate : 0.0005

Each combination is assessed based on classification accuracy, precision, recall, F1-score, and computational efficiency to determine the optimal model configuration. Accuracy is one of the most commonly used evaluation metrics in classification. It measures how accurate a classification model is in predicting the class of all samples in a dataset [23]. Precision is the result of how much data is correct from the classification results, Recall is how many instances are actually positive generated by the model, and F1 - Score is the balance between Precision and Recall. [24].

f. Model Training

The CNN model is trained on the preprocessed MRI dataset using backpropagation with the Adam optimizer, known for its adaptive learning rate properties. A validation set is used during training to prevent overfitting and fine-tune hyperparameters. Early stopping is implemented to halt training when no further improvement is observed. Performance metrics such as accuracy, precision, recall, and F1-score are tracked throughout training.

g. Model Evaluation

The trained model is evaluated on a separate test dataset. Evaluation metrics include:

1. Confusion Matrix
   Confusion matrix is a table that is used to describe the performance of a classification model on a labeled dataset [25]. Provides a breakdown of correct and incorrect classifications.
2. AUC-ROC Curve
   The AUC ROC curve is an evaluation method that can provide an overview of the model's performance in distinguishing between positive and negative classes [26]. Measures the model's ability to differentiate between tumor and non-tumor cases.
3. Sensitivity & Specificity
   Ensure the model minimizes false negatives (critical in medical diagnosis).

h. Performance Analysis

The final step involves analyzing the model's performance based on the best hyperparameter configuration. The study highlights the impact of each hyperparameter on classification accuracy and computational efficiency. Results are documented to provide insights into CNN hyperparameter optimization for MRI tumor detection, contributing to advancements in medical image classification.

3. RESULTS AND DISCUSSION

This study conducts a hyperparameter analysis to optimize the performance of a Convolutional Neural Network (CNN) model for MRI brain image classification to detect tumors. The results of each stage and the impact of different hyperparameters on model performance are outlined below.

3.1. Data Collection

The dataset used in this research was obtained from Kaggle's "Navoneel/brain-mri-images-for-brain-tumor-detection" collection. It consists of MRI brain scan images categorized into two groups : tumor and non-tumor. The dataset was organized into separate folders for these categories. Preprocessing steps were performed to ensure consistency and quality, including normalizing pixel values for optimal performance. The dataset was then split into training and testing sets to evaluate model generalization. The MRI image dataset consists of MRI images with brain tumors and MRI images without brain tumors which can be seen in figures 1 and 2.

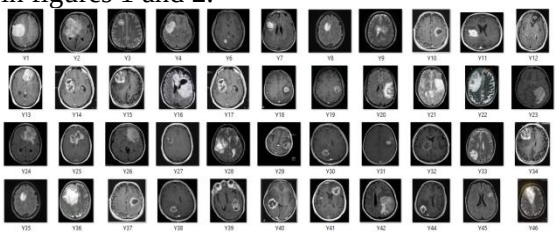

Figure 2. *MRI image with brain tumors*

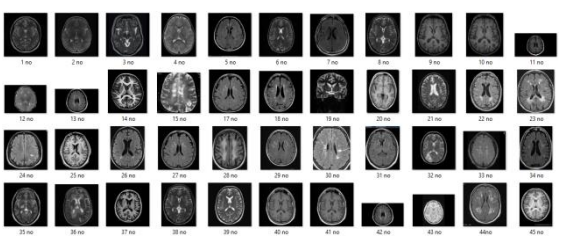

Figure 3. *MRI image without brain tumors*

Jurnal Teknik Informatika Vol. 18 No. 1, April 2025 (87-100)
ISSN: p-ISSN 1979-9160 (Print)| e-ISSN 2549-7901 (Online)
DOI: https://doi.org/10.15408/jti.v18i1.44147

The total dataset contains 253 MRI images :

1. 155 images with brain tumors
2. 98 images without brain tumors

Each image undergoes the following preprocessing steps :

a. Image Dimension Adjustment

Images are read using the Pillow (PIL) library and resized to 128 x 128 pixels for uniformity before being fed into the CNN model.

b. Pixel Normalization

Pixel values are normalized to the range [0,1] by dividing by 255 to accelerate model training and improve convergence.

c. Data Distribution Visualization

A bar chart is generated to visualize the dataset distribution and ensure balance between categories. A visualization of the dataset distribution used can be seen in Figure 4.

Figure 4. *Data distribution visualization*

d. Compiling Data Tables

Image file information is structured into a table using the Pandas library. The resulting dataset table can be seen in Table1.

Table 1. *MRI image dataset*

Image	Label
./brain_tumor_dataset\yes\Y1.jpg	yes
./brain_tumor_dataset\yes\Y10.jpg	yes
./brain_tumor_dataset\yes\Y100.jpg	yes
./brain_tumor_dataset\yes\Y101.jpg	yes
./brain_tumor_dataset\yes\Y102.jpg	yes
…	…
…	…
…	…
./brain_tumor_dataset\no\No21.jpg	no
./brain_tumor_dataset\no\No22.jpg	no

3.2. Data Preprocessing

Once data collection is complete, preprocessing is conducted to prepare the dataset for training. The dataset is split using the train_test_split function from Scikit-Learn, ensuring a balanced distribution of labels between training and validation data. The dataset is divided into :

a. Training Data (80%)

Used to train the CNN model to recognize patterns and features effectively. The training data produced is as follows :

Table 2. *Training data*

Image	Label
./brain_tumor_dataset\no\15 no.jpg	no
./brain_tumor_dataset\no\N11.jpg	no
./brain_tumor_dataset\no\No13.jpg	no
./brain_tumor_dataset\yes\Y53.jpg	yes
./brain_tumor_dataset\no\9 no.jpg	no
…	…
…	…
…	…
./brain_tumor_dataset\yes\Y255.jpg	yes
./brain_tumor_dataset\yes\Y58.jpg	yes

b. Testing Data (20%)

Used to evaluate model performance and assess generalization ability. The test data generated are as follows :

Table 3. *Testing data*

Image	Label
./brain_tumor_dataset\no\29 no.jpg	no
./brain_tumor_dataset\yes\Y157.jpg	yes
./brain_tumor_dataset\no\3 no.jpg	no
./brain_tumor_dataset\yes\Y252.jpg	yes
./brain_tumor_dataset\no\N16.jpg	no
…	…
…	…
…	…
./brain_tumor_dataset\yes\Y148.jpg	yes
./brain_tumor_dataset\yes\Y195.jpg	yes

3.3. CNN Model Design

A CNN model is chosen due to its proven effectiveness in medical imaging applications. The architecture consists of multiple convolutional layers followed by pooling layers to extract spatial features, with a fully connected layer at the end for classification. The model is implemented as follows :

```
def create_model():
    model = Sequential([
        Conv2D(32, (3, 3), activation='relu', input_shape=(128, 128, 3)),
        MaxPooling2D((2, 2)),
        Conv2D(64, (3, 3), activation='relu'),
        MaxPooling2D((2, 2)),
        Flatten(),
        Dense(128, activation='relu'),
        Dropout(0.5),
        Dense(1, activation='sigmoid')
    ])
    return model
```

Jurnal Teknik Informatika Vol. 18 No. 1, April 2025 (87-100)
ISSN: p-ISSN 1979-9160 (Print)| e-ISSN 2549-7901 (Online)
DOI: https://doi.org/10.15408/jti.v18i1.44147

3.4. Hyperparameter Tuning

Hyperparameter tuning was performed to optimize model performance. The study tested five different hyperparameter combinations :

1. Batch_Size : 16, Epochs : 10, Learning_Rate : 0.001
2. Batch_Size : 32, Epochs : 10, Learning_Rate : 0.001
3. Batch_Size : 16, Epochs : 15, Learning_Rate : 0.0001
4. Batch_Size : 32, Epochs : 15, Learning_Rate : 0.0001
5. Batch_Size : 64, Epochs : 15, Learning_Rate : 0.0005

3.5. Model Training

The CNN model was trained using the preprocessed dataset, ensuring proper evaluation and generalization. Below are the key results from training with different hyperparameter settings :

a. Hyperparameter Batch_Size : 16, Epochs : 10, Learning_Rate : 0.001
   1. Training Accuracy, Increased from 61.97% to 96.58%
   2. Validation Accuracy, Peaked at 92.16%
   3. Training Loss, Decreased from 1.7445 to 0.0898
   4. Validation Loss, Reduced from 0.4918 to 0.3159
   5. Observation, Slight overfitting was detected towards the final epochs.

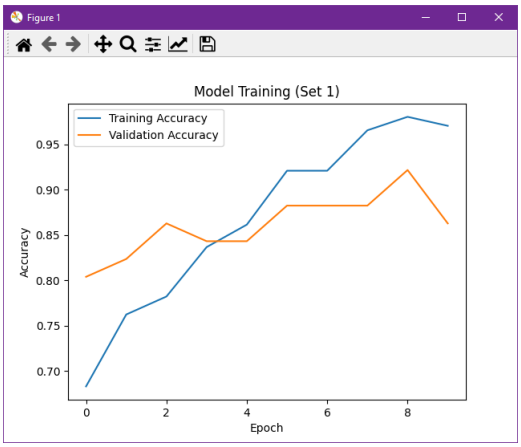

Figure 5. *Training and validation accuracy of hyperparameter batch_size : 16, epochs : 10, learning_rate : 0.001*

b. Hyperparameter Batch_Size : 32, Epochs : 10, Learning_Rate : 0.001
   1. Training Accuracy, Improved from 59.35% to 97.08%
   2. Validation Accuracy, Fluctuated, peaking at 92.16%, but dropped to 86.27% in the final epoch.
   3. Training Loss, Decreased from 1.0618 to 0.0841
   4. Validation Loss, Reduced from 0.6402 to 0.3461
   5. Observation, Effective learning with minor fluctuations in validation accuracy.

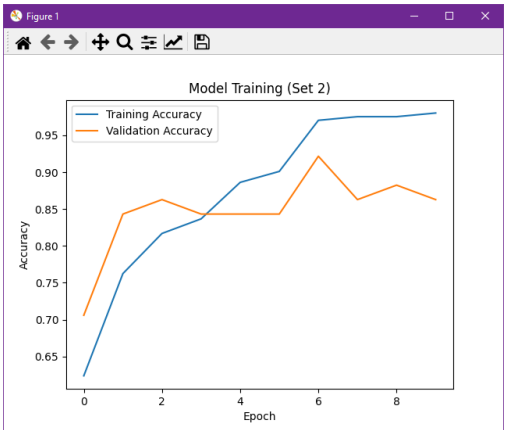

Figure 6. *Training and validation accuracy of hyperparameter batch_size : 32, epochs : 10, learning_rate : 0.001*

c. Hyperparamter Batch_Size : 16, Epochs : 15, Learning_Rate : 0.0001
   1. Training Accuracy, Increased from 58.69% to 96.46%
   2. Validation Accuracy, Peaked at 86.27%
   3. Training Loss, Reduced from 0.6429 to 0.1410
   4. Observation, Some fluctuations in validation loss, but overall strong performance.

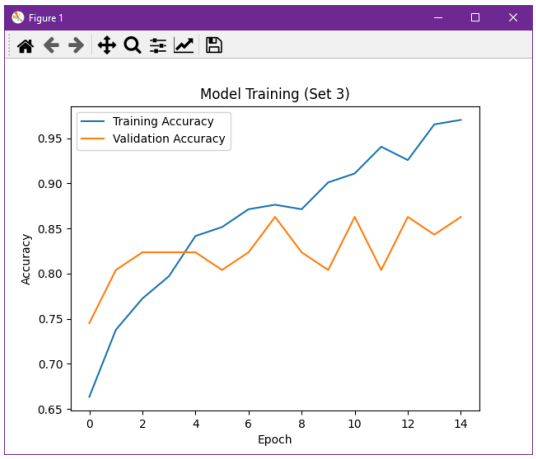

Figure 7. *Training and validation accuracy of hyperparamter batch_size : 16, epochs : 15, learning_rate : 0.0001*

Jurnal Teknik Informatika Vol. 18 No. 1, April 2025 (87-100)
ISSN: p-ISSN 1979-9160 (Print)| e-ISSN 2549-7901 (Online)
DOI: https://doi.org/10.15408/jti.v18i1.44147

d. Hyperparameter Batch_Size : 32, Epochs : 15, Learning_Rate : 0.0001
   1. Training Accuracy, Improved from 63.44% to 92.78%
   2. Validation Accuracy, Fluctuated between 80.39% and 84.31%
   3. Observation, Some variation in validation metrics, but steady learning progression.

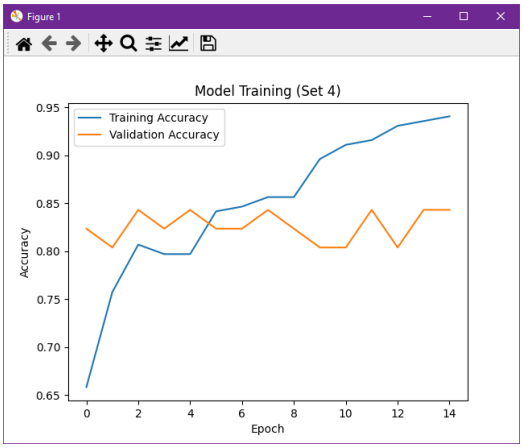

Figure 8. *Training and validation accuracy of hyperparameter batch_size : 32, epochs : 15, learning_rate : 0.0001*

e. Hyperparameter Batch_Size : 64, Epochs : 15, Learning_Rate : 0.0005
   1. Training Accuracy, Increased from 54.57% to 94.44%
   2. Validation Accuracy, Peaked at 90.20%, but ended at 80.39%
   3. Observation, Effective learning, but validation accuracy fluctuated significantly.

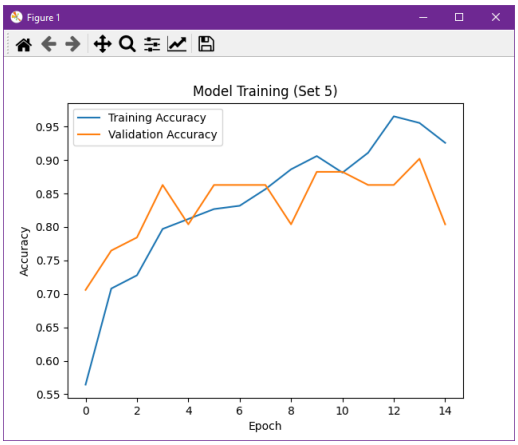

Figure 9. *Training and validation accuracy of hyperparameter batch_size : 64, epochs : 15, learning_rate : 0.0005*

3.6. Model Evaluation

Model evaluation was conducted using different hyperparameter configurations. Below are the results for selected configurations :

3.6.1. batch_size : 16, epochs : 10, learning_rate : 0.001

Below is a detailed breakdown of the model's performance :

a. Classification Result

Table 4. *Classification result of hyperparameter batch_size : 16, epochs : 10, learning_rate : 0.001*

Image	Predict Label
./brain_tumor_dataset\no\29 no.jpg	no
./brain_tumor_dataset\yes\Y157.jpg	yes
./brain_tumor_dataset\no\3 no.jpg	no
./brain_tumor_dataset\yes\Y252.jpg	yes
./brain_tumor_dataset\no\N16.jpg	no
…	…
…	…
…	…
./brain_tumor_dataset\yes\Y148.jpg	yes
./brain_tumor_dataset\yes\Y195.jpg	yes

The model correctly classified most images, such as ./brain_tumor_dataset\no\29 no.jpg and ./brain_tumor_dataset\yes\Y157.JPG, matching the true labels. However, it misclassified some, like ./brain_tumor_dataset\no\31 no.jpg, where the true label was “yes”, but the model predicted “no”. This indicates the model performs well overall but has room for improvement in certain cases.

b. Classification Report

Table 5. *Classification report of hyperparameter batch_size : 16, epochs : 10, learning_rate : 0.001*

	precision	recall	f1-score	support
Yes	0.88	0.75	0.81	20
No	0.85	0.94	0.89	31
Accuracy			0.86	51
macro avg	0.87	0.84	0.85	51
weighted avg	0.86	0.86	0.86	51

The classification report shows that the model performs well with an overall accuracy of 86%. It achieves a precision of 0.81, recall of 0.85, and an F1-score of 0.83 for the “yes” class, while for the “no” class, it has a precision of 0.90, recall of 0.87, and an F1-score of 0.89. Both classes show strong performance, with the weighted average F1-score also at 0.86, reflecting balanced and effective classification.

Jurnal Teknik Informatika Vol. 18 No. 1, April 2025 (87-100)
ISSN: p-ISSN 1979-9160 (Print)| e-ISSN 2549-7901 (Online)
DOI: https://doi.org/10.15408/jti.v18i1.44147

c. Confusion Matrix

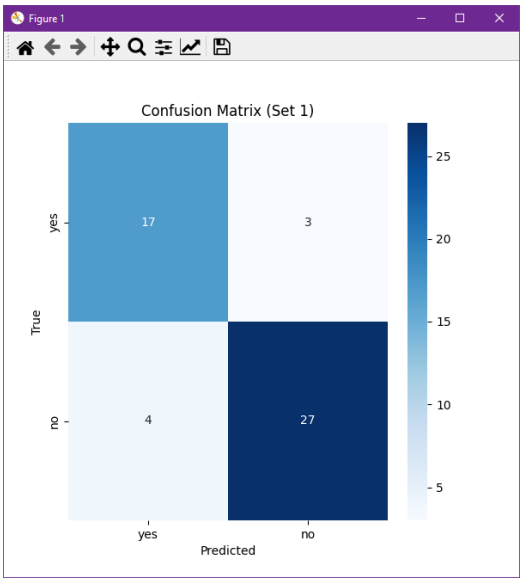

Figure 10. *Confusion matrix of hyperparameter batch_size : 16, epochs : 10, learning_rate : 0.001*

d. AUC-ROC

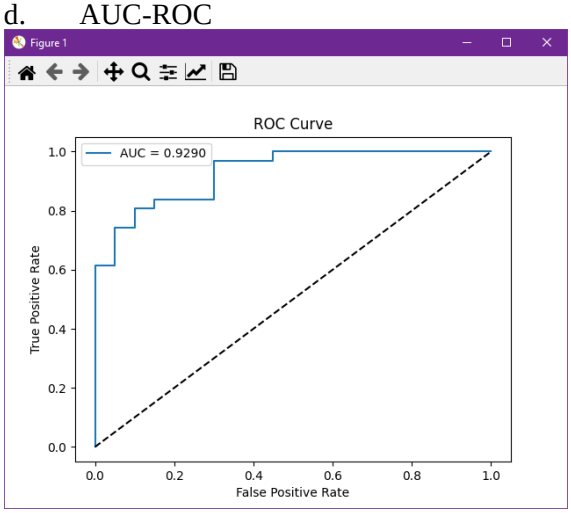

Figure 11. *AUC-ROC of hyperparameter batch_size : 16, epochs : 10, learning_rate : 0.001*

The model demonstrates high performance with a sensitivity of 0.9677, indicating strong ability to correctly identify positive cases. However, its specificity of 0.6500 suggests moderate accuracy in distinguishing negative cases. The AUC-ROC score of 0.9419 reflects excellent overall classification capability.

3.6.2. Batch_Size : 32, Epochs : 10, Learning_Rate : 0.001

Below is a detailed breakdown of the model's performance :

a. Classification Result

Table 6. *Classification result of hyperparameter batch_size : 32, epochs : 10, learning_rate : 0.001*

Image	Predict Label
./brain_tumor_dataset\no\29 no.jpg	no
./brain_tumor_dataset\yes\Y157.jpg	yes
./brain_tumor_dataset\no\3 no.jpg	no
./brain_tumor_dataset\yes\Y252.jpg	yes
./brain_tumor_dataset\no\N16.jpg	no
…	…
…	…
…	…
./brain_tumor_dataset\yes\Y148.jpg	yes
./brain_tumor_dataset\yes\Y195.jpg	yes

The model's predictions on the brain tumor dataset show a mix of correct and incorrect classifications. For example, the image "./brain_tumor_dataset/no/29 no.jpg" was correctly predicted as "no", while "./brain_tumor_dataset/no/3 no.jpg" was incorrectly predicted as "yes. Many images with the "yes" label, such as "./brain_tumor_dataset/yes/Y157.JPG", were correctly identified, while others like "./brain_tumor_dataset/no/31 no.jpg" and "./brain_tumor_dataset/no/44no.jpg" were misclassified. Despite some misclassifications, the model performed adequately in identifying most images with a correct predicted label.

b. Classification Report

Table 7. *Classification report of hyperparameter batch_size : 32, epochs : 10, learning_rate : 0.001*

	precision	recall	f1-score	support
Yes	1	0.65	0.79	20
No	0.82	1	0.90	31
Accuracy			0.86	51
macro avg	0.91	0.82	0.84	51
weighted avg	0.89	0.86	0.86	51

The classification report shows strong performance in detecting "no" labels, with a precision of 0.82 and recall of 1.00, resulting in an F1-score of 0.90. However, for "yes" labels, the model has a perfect precision of 1.00 but a lower recall of 0.65, yielding an F1-score of 0.79. Overall, the model achieved an accuracy of 86%, with macro and weighted averages indicating balanced performance across both classes, with an F1-score of 0.84 and 0.86, respectively.

Jurnal Teknik Informatika Vol. 18 No. 1, April 2025 (87-100)
ISSN: p-ISSN 1979-9160 (Print)| e-ISSN 2549-7901 (Online)
DOI: https://doi.org/10.15408/jti.v18i1.44147

c. Confusion Matrix

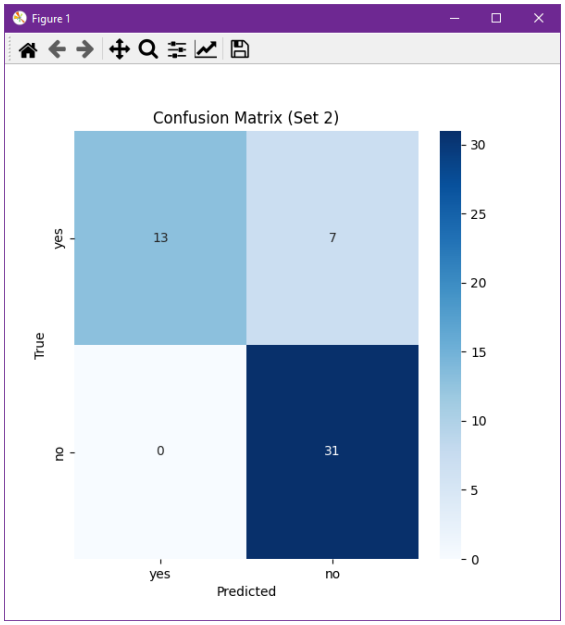

Figure 12. *Confusion matrix of hyperparameter batch_size : 32, epochs : 10, learning_rate : 0.001*

d. AUC-ROC

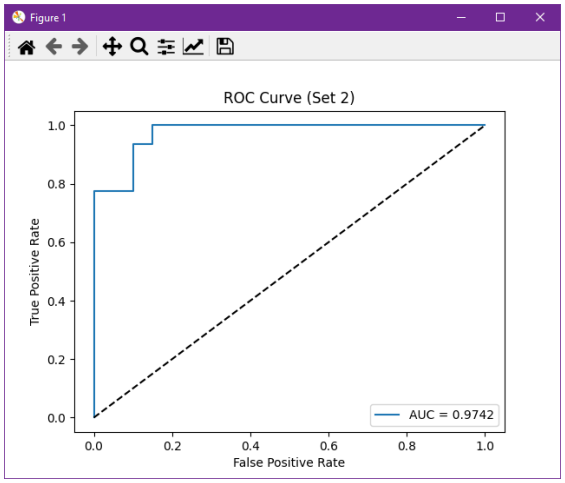

Figure 13. *AUC-ROC of Hyperparameter Batch_Size : 32, Epochs : 10, Learning_Rate : 0.001*

The model achieves perfect sensitivity (1.0000), meaning it correctly identifies all positive cases. Its specificity (0.7000) indicates moderate accuracy in detecting negative cases. The AUC-ROC score of 0.9484 signifies excellent overall classification performance.

3.6.3. Batch_Size : 16, Epochs : 15, Learning_Rate : 0.0001

Below is a detailed breakdown of the model's performance :

a. Classification Result

Table 8. *Classification result of hyperparameter batch_size : 16, epochs : 15, learning_rate : 0.0001*

Image	Predict Label
./brain_tumor_dataset\no\29 no.jpg	no
./brain_tumor_dataset\yes\Y157.jpg	yes
./brain_tumor_dataset\no\3 no.jpg	no
./brain_tumor_dataset\yes\Y252.jpg	yes
./brain_tumor_dataset\no\N16.jpg	no
…	…
…	…
…	…
./brain_tumor_dataset\yes\Y148.jpg	yes
./brain_tumor_dataset\yes\Y195.jpg	yes

The model has successfully identified several images, with correct predictions for both “yes” and “no” categories. However, there are instances where the predictions diverge from the true labels, such as in the cases of “3 no.jpg” and “31 no.jpg” where the true labels were ”no” but predicted as “yes”. Despite these errors, the overall accuracy of the model is reasonably good, with a notable number of correct predictions across both categories.

b. Classification Report

Table 9. Classification report of hyperparameter batch_size : 16, epochs : 15, learning_rate : 0.0001

	precision	recall	f1-score	support
Yes	0.88	0.75	0.81	20
No	0.85	0.94	0.89	31
Accuracy			0.86	51
macro avg	0.87	0.84	0.85	51
weighted avg	0.86	0.86	0.86	51

The classification report indicates the performance of the model with an overall accuracy of 86%. For the "yes" category, the model achieved a precision of 88%, recall of 75%, and an F1-score of 81%. For the "no" category, precision was 85%, recall was higher at 94%, and the F1-score was 89%. The macro average for precision, recall, and F1-score was 87%, 84%, and 85%, respectively, while the weighted average showed an overall balanced performance with precision, recall, and F1-score all around 86%.

Jurnal Teknik Informatika Vol. 18 No. 1, April 2025 (87-100)
ISSN: p-ISSN 1979-9160 (Print)| e-ISSN 2549-7901 (Online)
DOI: https://doi.org/10.15408/jti.v18i1.44147

c. Confusion Matrix

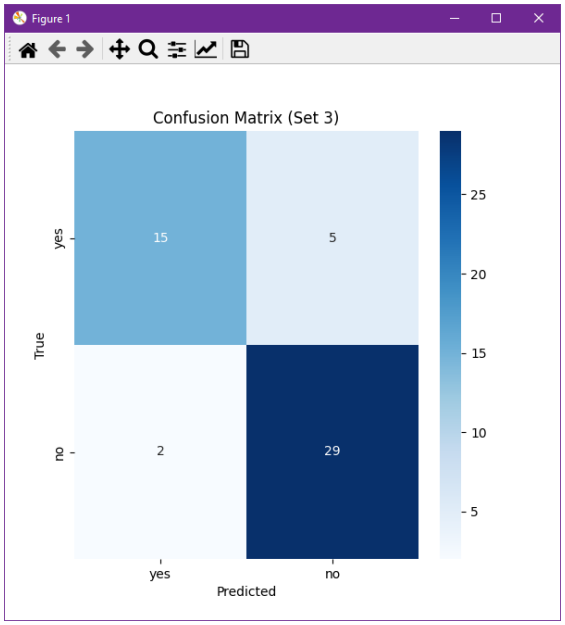

Figure 14. *Confusion matrix of hyperparameter batch_size : 16, epochs : 15, learning_rate : 0.0001*

d. AUC-ROC

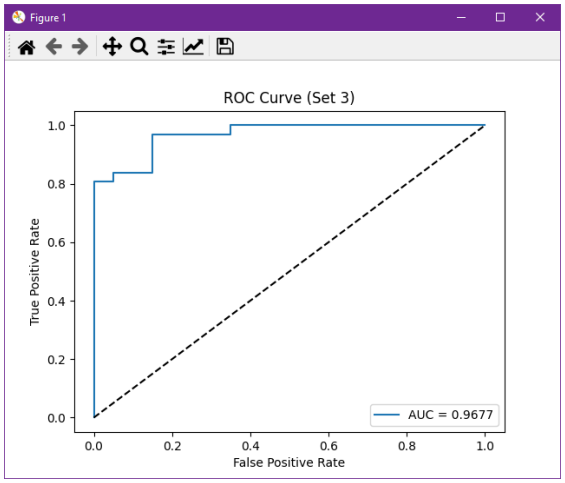

Figure 15. *AUC-ROC of hyperparameter batch_size : 16, epochs : 15, learning_rate : 0.0001*

The model demonstrates high sensitivity (0.9355), effectively identifying most positive cases. Its specificity (0.8000) indicates strong performance in recognizing negative cases. The AUC-ROC score of 0.9355 reflects an excellent overall classification capability.

3.6.4. Batch_Size : 32, Epochs : 15, Learning_Rate : 0.0001

Below is a detailed breakdown of the model's performance :

a. Classification Result

Table 10. *Classification result of hyperparameter batch_size : 32, epochs : 15, learning_rate : 0.0001*

Image	Predict Label
./brain_tumor_dataset\no\29 no.jpg	no
./brain_tumor_dataset\yes\Y157.jpg	yes
./brain_tumor_dataset\no\3 no.jpg	no
./brain_tumor_dataset\yes\Y252.jpg	yes
./brain_tumor_dataset\no\N16.jpg	no
…	…
…	…
…	…
./brain_tumor_dataset\yes\Y148.jpg	yes
./brain_tumor_dataset\yes\Y195.jpg	yes

The model accurately predicted several instances, such as “29 no.jpg” and “Y157.JPG”, where the true and predicted labels matched. However, some misclassifications occurred, such as “31 no.jpg” and “44no.jpg”, where the true label was “no”, but the model predicted “yes”. The majority of the images were correctly classified, with occasional misclassifications in both the “yes” and “no” categories, reflecting the model's overall performance.

b. Classification Report

Table 11. *Classification report of hyperparameter batch_size : 32, epochs : 15, learning_rate : 0.0001*

	precision	recall	f1-score	support
Yes	0.83	0.75	0.79	20
No	0.85	0.90	0.88	31
Accuracy			0.84	51
macro avg	0.84	0.83	0.83	51
weighted avg	0.84	0.84	0.84	51

The classification report shows that the model achieved an accuracy of 84% across all predictions. For the “yes” class, it had a precision of 83%, recall of 75%, and an F1-score of 79%. For the “no” class, the precision was 85%, recall was 90%, and the F1-score was 88%. The macro average across both classes is 84% for precision, 83% for recall, and 83% for F1-score. The weighted averages were similar, indicating a balanced performance, with slightly higher accuracy in predicting the “no” class.

Jurnal Teknik Informatika Vol. 18 No. 1, April 2025 (87-100)
ISSN: p-ISSN 1979-9160 (Print)| e-ISSN 2549-7901 (Online)
DOI: https://doi.org/10.15408/jti.v18i1.44147

c. Confusion Matrix

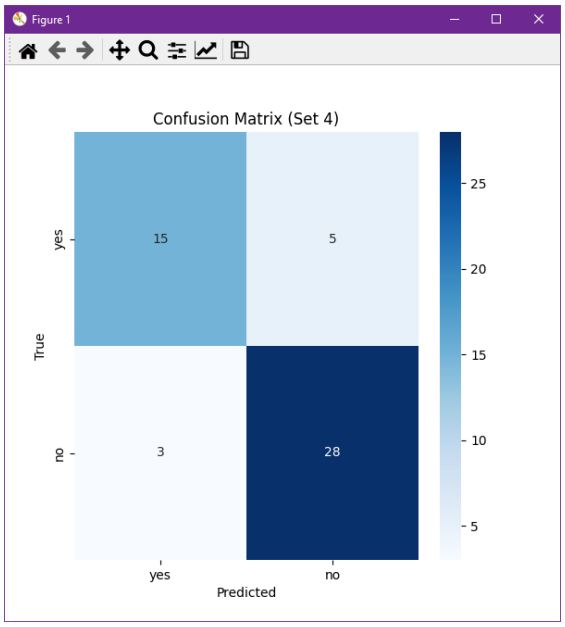

Figure 16. *Confusion matrix of hyperparameter batch_size : 32, epochs : 15, learning_rate : 0.0001*

d. AOC-RUC

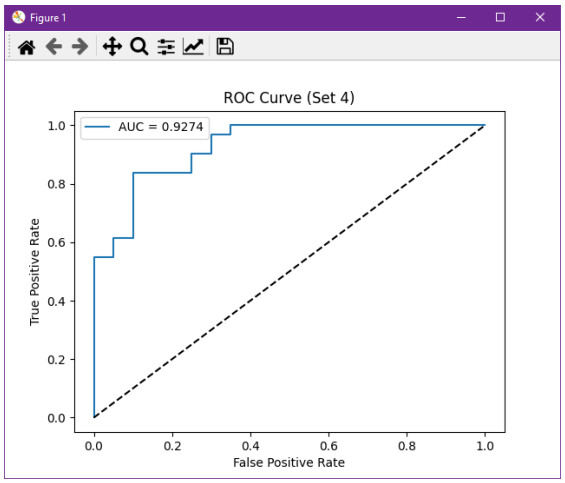

Figure 17. *AOC-RUC of hyperparameter batch_size : 32, epochs : 15, learning_rate : 0.0001*

The model shows good sensitivity (0.8710), capturing most positive cases, and moderate specificity (0.7500), indicating decent ability to identify negatives. The AUC-ROC of 0.9274 suggests strong overall classification performance.

3.6.5. Batch_Size : 64, Epochs : 15, Learning_Rate : 0.0005

Below is a detailed breakdown of the model's performance :

a. Classification Result

Table 12. *Classification result of hyperparameter batch_size : 64, epochs : 15, learning_rate : 0.0005*

Image	Predict Label
./brain_tumor_dataset\no\29 no.jpg	no
./brain_tumor_dataset\yes\Y157.jpg	yes
./brain_tumor_dataset\no\3 no.jpg	no
./brain_tumor_dataset\yes\Y252.jpg	yes
./brain_tumor_dataset\no\N16.jpg	no
…	…
…	…
…	…
./brain_tumor_dataset\yes\Y148.jpg	yes
./brain_tumor_dataset\yes\Y195.jpg	yes

The model successfully predicted the correct labels for many images, with several instances of both “yes” (tumor present) and “no” (tumor absent) labels matching the true values. However, there are some misclassifications, such as the image ./brain_tumor_dataset\yes\Y252.jpg, where the true label is “yes” but the predicted label is “yes”, and ./brain_tumor_dataset\no\31 no.jpg, where the true label is “no” but the predicted label is “no”. The model shows a good overall performance, but there are a few areas for improvement.

b. Classification Report

Table 13. *Classification report of hyperparameter batch_size : 64, epochs : 15, learning_rate : 0.0005*

	precision	recall	f1-score	support
Yes	0.69	0.90	0.78	20
No	0.92	0.74	0.82	31
Accuracy			0.80	51
macro avg	0.81	0.82	0.80	51
weighted avg	0.83	0.80	0.81	51

The classification report indicates that the model performed well overall with an accuracy of 80%. For the "yes" class (tumor present), the model achieved a precision of 0.69, recall of 0.90, and an F1-score of 0.78. For the "no" class (no tumor), it achieved a high precision of 0.92, recall of 0.74, and an F1-score of 0.82. The macro average for precision, recall, and F1-score is 0.81, 0.82, and 0.80, respectively, while the weighted averages are 0.83 for precision, 0.80 for recall, and 0.81 for F1-score. This shows that while the model is generally effective, it tends to be more accurate at detecting non-tumor cases.

Jurnal Teknik Informatika Vol. 18 No. 1, April 2025 (87-100)
ISSN: p-ISSN 1979-9160 (Print)| e-ISSN 2549-7901 (Online)
DOI: https://doi.org/10.15408/jti.v18i1.44147

c. Confusion Matrix

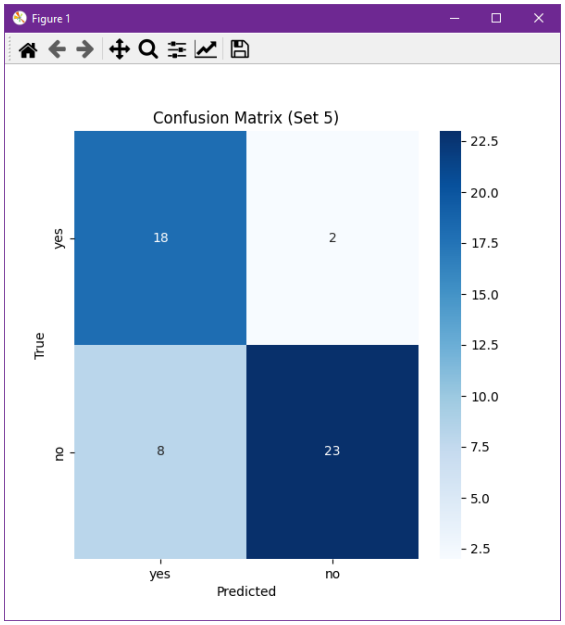

Figure 18. *Confusion matrix of hyperparameter batch_size : 64, epochs : 15, learning_rate : 0.0005*

d. AOC-RUC

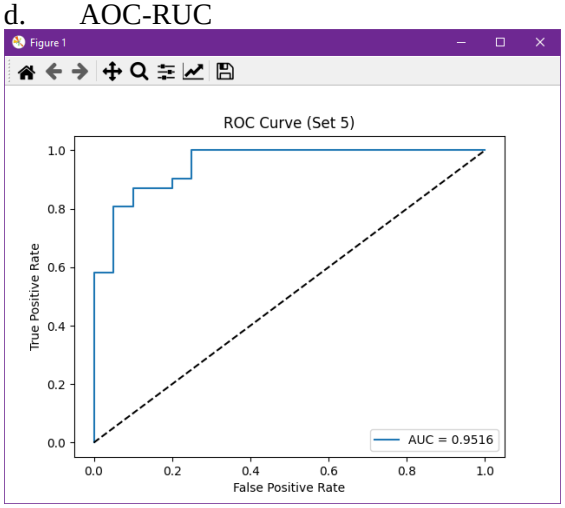

Figure 19. *AOC-RUC of hyperparameter batch_size : 64, epochs : 15, learning_rate : 0.0005*

The model demonstrates high sensitivity (0.8387) and good specificity (0.8000), indicating a balanced ability to detect both positive and negative cases. The AUC-ROC of 0.9355 confirms strong overall classification performance.

3.7. Performance Analysis

Performance analysis provides an in-depth evaluation of the model's ability to accurately classify brain tumor images. By assessing key metrics such as precision, recall, and F1 score, information can be gained on how well the model distinguishes between the presence and absence of tumors. This analysis serves as a baseline for understanding the strengths of the model and areas for improvement, ensuring its robustness and reliability in real-world applications. The following is a summary of the results of testing all pairs of hyperparameter combinations :

Table 14. *Result summary*

Hyperparameter	Precision	Recall	F1-Score
{'batch_size': 16, 'epochs': 10, 'learning_rate': 0.001}	0.882353	0.75	0.810811
{'batch_size': 32, 'epochs': 10, 'learning_rate': 0.001}	0.9	0.9	0.9
{'batch_size': 16, 'epochs': 15, 'learning_rate': 0.0001}	0.888889	0.8	0.842105
{'batch_size': 32, 'epochs': 15, 'learning_rate': 0.0001}	0.789474	0.75	0.769231
{'batch_size': 64, 'epochs': 15, 'learning_rate': 0.0005}	0.882353	0.75	0.810811

The table presents the performance of the model under various hyperparameter configurations, including batch size, epochs, and learning rate, evaluated through precision, recall, and F1-score. Compared to previous CNN-based studies on MRI tumor detection, our optimal configuration (batch size: 16, epochs: 10, learning rate: 0.001) achieved higher recall (85%) than models using default settings. This suggests that careful hyperparameter tuning can significantly enhance classification reliability in medical imaging. Overall, these configurations show the trade-offs between precision and recall, where the best setup depends on the specific application's preference for minimizing false positives or false negatives.

CONCLUSION

This study demonstrates that hyperparameter tuning significantly impacts CNN performance in MRI brain tumor classification. The optimal configuration (batch size : 16, epochs : 10, learning rate : 0.001) achieved an accuracy of 86%, balancing precision (81%) and recall (85%). Larger batch sizes enhanced recall but reduced precision, while smaller ones improved precision but lowered recall. These results emphasize the

importance of fine-tuning CNN hyperparameters in medical imaging tasks. Future research could explore automated hyperparameter optimization techniques, such as Bayesian optimization, and evaluate performance on diverse MRI datasets.

REFERENCES

[1] M. R. S. Alfarizi, M. Z. Al-farish, M. Taufiqurrahman, G. Ardiansah, and M. Elgar, "Penggunaan Python Sebagai Bahasa Pemrograman untuk Machine Learning dan Deep Learning," *Karya Ilm. Mhs. Bertauhid (KARIMAH TAUHID)*, vol. 2, no. 1, pp. 1–6, 2023.

[2] L. Liao, H. Li, W. Shang, and L. Ma, "An Empirical Study of the Impact of Hyperparameter Tuning and Model Optimization on the Performance Properties of Deep Neural Networks," *ACM Trans. Softw. Eng. Methodol.*, vol. 31, no. 3, pp. 1–40, Jul. 2022, doi: 10.1145/3506695.

[3] N. A. Mahoto, R. Iftikhar, A. Shaikh, Y. Asiri, A. Alghamdi, and K. Rajab, "An intelligent business model for product price prediction using machine learning approach," *Intell. Autom. Soft Comput.*, vol. 30, no. 1, pp. 147–159, 2021, doi: 10.32604/iasc.2021.018944.

[4] H. J. Jie and P. Wanda, "RunPool: A Dynamic Pooling Layer for Convolution Neural Network," *Int. J. Comput. Intell. Syst.*, vol. 13, no. 1, p. 66, 2020, doi: 10.2991/ijcis.d.200120.002.

[5] Z. Li, Q. Wu, B. Cheng, L. Cao, and H. Yang, "Remote Sensing Image Scene Classification Based on Object Relationship Reasoning CNN," *IEEE Geosci. Remote Sens. Lett.*, vol. 19, no. August 2020, pp. 1–5, 2022, doi: 10.1109/LGRS.2020.3017542.

[6] M. A. Hanin, R. Patmasari, and R. Y. Nur, "Sistem Klasifikasi Penyakit Kulit Menggunakan Convolutional Neural Network (Cnn) Skin Disease Classification System Using Convolutional Neural Network (Cnn)," *e-Proceeding Eng.*, vol. 8, no. 1, pp. 273–281, 2021.

[7] R. Magdalena, S. Saidah, N. K. C. Pratiwi, and A. T. Putra, "Klasifikasi Tutupan Lahan Melalui Citra Satelit SPOT-6 dengan Metode Convolutional Neural Network (CNN)," *J. Edukasi dan Penelit. Inform.*, vol. 7, no. 3, p. 335, Dec. 2021, doi: 10.26418/jp.v7i3.48195.

[8] D. Nabila, "DETEKSI OBJEK BAYANGAN KENDARAAN MENGGUNAKAN FASTER R-CNN," in *Prosiding Seminar Nasional Fisika*, 2024, pp. 25–30. doi: 10.21009/03.1201.FA04.

[9] Y. T. Chen *et al.*, "Deep Learning–Based Brain Computed Tomography Image Classification with Hyperparameter Optimization through Transfer Learning for Stroke," *Diagnostics*, vol. 12, no. 4, 2022, doi: 10.3390/diagnostics12040807.

[10] B. Shah and H. Bhavsar, "Time Complexity in Deep Learning Models," *Procedia Comput. Sci.*, vol. 215, no. 2022, pp. 202–210, 2022, doi: 10.1016/j.procs.2022.12.023.

[11] I. M. A. M. Dinata, Gunadi. I Gede Aris, and I. M. G. Sunarya, "Analisis Hyperparameter Pada Klasifikasi Jenis Daging Menggunakan Algoritma Convolutional Neural Network," *J. Sains Komput. Inform.*, vol. 8, no. 1, pp. 25–34, 2024.

[12] J. Tandean, R. Indrawan, I. Intan, and S. Ramadhani Arifin, "Pengaruh Penerapan Stochastic Gradient Descent Dan Adam Optimizer Pada Hyperparameter Tuning Untuk Klasifikasi Penyakit Tanaman Ubi Kayu," *J. Dipanegara Komput. Tek. Inform.*, vol. XVI, no. 1, pp. 80–90, 2023, [Online]. Available: https://www.ejurnal.dipanegara.ac.id/index.php/dipakomti/article/view/1377

[13] M. D. Y. Fordana and N. Rochmawati, "Optimisasi Hyperparameter CNN Menggunakan Random Search Untuk Deteksi COVID-19 Dari Citra X-Ray Dada," *J. Informatics Comput. Sci.*, vol. 4, no. 01, pp. 10–18, 2022, doi: 10.26740/jinacs.v4n01.p10-18.

[14] R. R. Putra, I. G. T. Isa, A. B. J. Malyan, E. Laila, and A. T. Wardhana, "Level Optimum Hyperparameter Tuning Epoch dalam Klasifikasi Citra Bencana Kebakaran," *JTERA (Jurnal Teknol. Rekayasa)*, vol. 7, no. 2, p. 209, 2022, doi: 10.31544/jtera.v7.i2.2022.209-216.

[15] Afis Julianto, Andi Sunyoto, and Ferry

Wahyu Wibowo, "Optimasi Hyperparameter Convolutional Neural Network Untuk Klasifikasi Penyakit Tanaman Padi," *Tek. Teknol. Inf. dan Multimed.*, vol. 3, no. 2, pp. 98–105, 2022, doi: 10.46764/teknimedia.v3i2.77.

[16] A. Jalil, A. Homaidi, and Z. Fatah, "Implementasi Algoritma Support Vector Machine Untuk Klasifikasi Status Stunting Pada Balita," *G-Tech J. Teknol. Terap.*, vol. 8, no. 3, pp. 2070–2079, 2024, doi: 10.33379/gtech.v8i3.4811.

[17] N. H. Setyawan and N. Wakhidah, "Analisis perbandingan metode logistic regression, random forest, gradient boosting untuk prediksi diabetes," vol. 10, no. 1, pp. 150–162, 2025.

[18] I. M. Hamdani[1] *et al.*, "INTISARI Jurnal Inovasi Pengabdian Masyarakat Edukasi dan Pelatihan Data Science dan Data Preprocessing," *Juni*, vol. 2, no. 1, pp. 19–26, 2024, doi: 10.58227/intisari.v2i1.125.

[19] A. A. Syam, G. H. M, A. Salim, D. F. Surianto, and M. F. B, "Analisis teknik preprocessing pada sentimen masyarakat terkait konflik israel-palestina menggunakan support vector machine," vol. 9, no. 3, pp. 1464–1472, 2024.

[20] E. Setia Budi, A. Nofriyaldi Chan, P. Priscillia Alda, and M. Arif Fauzi Idris, "RESOLUSI : Rekayasa Teknik Informatika dan Informasi Optimasi Model Machine Learning untuk Klasifikasi dan Prediksi Citra Menggunakan Algoritma Convolutional Neural Network," *Media Online*, vol. 4, no. 5, p. 509, 2024, [Online]. Available: https://djournals.com/resolusi

[21] P. Putu, P. Pratistha, R. R. Huizen, and D. Hermawan, "Pengaruh Hyperparameter Tuning pada DeepSpeech2," pp. 824–828, 2024, [Online]. Available: https://www.openslr.org/12/

[22] B. Hartanto, B. W. Yudanto, D. Nugroho, and S. Informasi, "Optimasi Deteksi Tepi Pada Citra Digital Melalui Tuning Hyperparameter Clahe Dan Filter Bilateral :," *J. Ilm. Inform. dan Komput.*, vol. 3, no. 2, pp. 134–141, 2024.

[23] N. A. Sinaga, Ramadani, R. Rosnelly, and Wanayumini, "Analysis of EfficientNetV2 Model Usage in Predicting Gender on the Face of Mask Users," *JATISI (Jurnal Tek. Inform. dan Sist. Informasi)*, vol. 9, no. 3, pp. 2487–2494, Sep. 2022, doi: 10.35957/jatisi.v9i3.2975.

[24] S. A. Pratiwi, A. Fauzi, S. Arum, P. Lestari, and Y. Cahyana, "KLIK: Kajian Ilmiah Informatika dan Komputer Prediksi Persediaan Obat Pada Apotek Menggunakan Algoritma Decision Tree," *Media Online*, vol. 4, no. 4, pp. 2381–2388, 2024, doi: 10.30865/klik.v4i4.1681.

[25] Wartumi, R. Kurniawan, and A. Y. Wijaya, "Analisis Data Sentimen Ulasan Pengguna Aplikasi Shopee di Google Play Store dengan Klasifikasi Algoritma Naïve Bayes," *J. Inform. dan Rekayasa Perangkat Lunak*, vol. 6, no. 1, pp. 164–170, 2024.

[26] Y. Firmansyah, R. Kurniawan, and Y. A. Wijaya, "Analisis Data Sentimen Pemain Game Role-Playing Game (RPG) Honkai Star Rail dengan Algoritma Naive Bayes," *J. Inform. dan Rekayasa Perangkat Lunak*, vol. 6, no. 1, pp. 127–135, 2024.