Brain Tumor Classification Using Deep Learning CNN Method

Justin Stephen, Nurfajriah Oktaviani, Geraldo Nathanael
[12345] Information System, Faculty of Engineering & Information
Universitas Multimedia Nusantara, Tangerang Banten 15810, Indonesia.

,

, [1]nurfajriah.oktaviani@student.umn.ac.id

***Abstrak* – Brain tumors are diseases that affect the central nervous system and have a significant impact on patients' health and quality of life. This research aims to improve brain tumor classification using Deep Learning technology, particularly Convolutional Neural Networks (CNN). This method utilizes MRI brain image datasets of three main tumor types: meningioma, glioma, and pituitary tumor. Data preprocessing involves converting images to grayscale, resizing, and applying filters to reduce noise. Subsequently, a CNN model is developed and evaluated using performance metrics such as accuracy, precision, recall, and F1-score. The results show that the CNN model can distinguish between different types of brain tumors with high sensitivity and specificity. In conclusion, Deep Learning technology, especially CNN, has great potential in improving brain tumor diagnosis, paving the way for more effective and personalized treatments.**

Keywords: *Brain tumor, Deep Learning, Convolutional Neural Networks, classification, medical imaging*

I. INTRODUCTION

A. Background and Objectives

Brain tumor is one of the diseases that affect the central nervous system and has a significant impact on the health and quality of life of patients [1]. In an effort to understand more about brain tumors and improve diagnosis and proper treatment, research in this area has become very important [2]. One important aspect of the study was the identification and classification of different types of brain tumors, such as meningiomas, gliomas, and pituitary tumors.

Research conducted by Cheng et al. (2015, 2016) has made important contributions in this field by using advanced image processing and data analysis technologies to improve the performance of brain tumor classification. The research shows how image processing, such as T1 contrast ballast on brain scans, can provide important information about the structural characteristics of tumors [3]. However, there is still room for improvement, particularly in the development of models that can more accurately distinguish between types of brain tumors [4].

Given the high level of complexity in the classification of brain tumors and the importance of timely diagnosis, this study aims to address some of these challenges by utilizing Deep Learning technologies, particularly Convolutional Neural Networks (CNN) [5]. CNN offers a number of advantages over traditional image processing methods, including the ability to automatically extract and study important features from image data, allowing for more accurate and efficient classification[6].

In addition, research by Litjens et al. (2017) and Shen et al. (2019) has shown how CNN can improve the accuracy of diagnosis of various medical conditions through image analysis, with brain tumor detection and classification being one of the most promising applications[7]. Both studies confirm the great potential of using CNN in improving brain tumor diagnosis, offering significant improvements in sensitivity and specificity compared to traditional methods [8].

This, the urgency of this research is driven by the need to overcome the limitations of current diagnostic methods and leverage technological advances in deep learning to develop more accurate and effective solutions in brain tumor classification [9]. The results of this study will hopefully provide a solid foundation for the development of better diagnostic techniques and more effective treatment strategies for patients with brain tumors, while paving the way for wider applications of CNN in medicine and diagnostics[10].

The primary objective of this research is to develop a sophisticated brain tumor classification model that can accurately differentiate between meningiomas, gliomas, and pituitary tumors. This will be achieved by leveraging advanced convolutional neural network (CNN) architectures. A significant focus of the study is on the evaluation and refinement of feature extraction processes and image processing techniques within these CNN models. By optimizing these elements, the research aims to improve the diagnostic accuracy of brain tumor classification, as measured by key performance metrics such as accuracy, precision, recall, and F1-score. Additionally, the study seeks to assess the effectiveness of the proposed methods using substantial datasets, ensuring that the improvements in diagnostic precision are statistically significant and practically applicable in clinical settings. Through these objectives, the research aspires to contribute valuable advances to the field of medical imaging and brain tumor diagnosis.

B. Literature Review

a) Basic Concepts and Applications of Deep Learning in Medical

Deep learning is a branch of machine learning, utilizing neural networks to model and process complex data at unprecedented scale. Using architecture inspired by the structure and function of the human brain, deep learning has revolutionized many fields, including medicine, with its unique ability to capture, analyze, and interpret medical image data.

qIn medicine, deep learning applications are especially prominent in medical image analysis. Through the use of complex models, this technology has opened up new avenues in the identification, classification, and segmentation of various types of tumors, making a significant contribution to the diagnosis and planning of patient care. One of the main advantages of deep learning is its ability to automatically extract relevant features from medical images without requiring manual feature extraction which usually requires expert intervention and is often subjective.

qResearch by Litjens et al. (2017) underscores the potential of deep learning in improving the accuracy of diagnosis of various medical conditions through image analysis. Research by Shen et al. (2019) emphasizes the important role of Convolutional Neural Networks (CNNs), a subclass of deep learning, in the detection and classification of brain tumors, showing significant improvements in sensitivity and specificity compared to traditional methods.

b) Model of Convolutional Neural Networks (CNN)

CNNs (Convolutional Neural Networks) are specialized architectures of deep neural networks that are highly effective for image processing and computer vision applications. By utilizing a structure consisting of convolution, pooling, and fully connected layers, CNNs are able to extract important features from imagery and classify them based on those features.

Input Layer

The Input Layer accepts input images with dimensions WxHxD, where W is width, H is height, and D is depth (for example, 3 for RGB images).

Convolution Layer

Performs convolution operations between filter F with fx size fx D and input image or feature map from the previous layer. If we have a K filter, then the output of this layer is W' H' x K where W' and H' are the width and height of the feature map after convolution. Convolution operations can be calculated as:

$$G(x, y) = \Sigma(i, j)\ F(i, j) * I(x - i, y - j)$$

where G(x, y) is the convolution output at position (x, y), F(i, j) is the value in the filter at position (i, j), and I(xi, yj) is the intensity value in the input image or feature map at the appropriate position.

Pooling Layer

The Pooling layer aims to reduce the dimensions of the feature map by implementing pooling operations such as max pooling or average pooling. For example, for max pooling with a window size of 2 x 2, the operations performed are:

$$P(x,y) = \max(i,j) \in Wx,y\ I(i,j)$$

where P(x, y) is the pooling output value at position (x, y) and Wry is the pooling window covering position (x, y).

Fully Connected (FC) Layer

Each neuron in this layer is connected to all activations in the previous layer. If we have M neurons in the FC layer and N activation from the previous layer, then the output of k neurons can be calculated as:

$$Ok = \sigma\ (\Sigma\ N\ n = 1\ wk,n * xn + bk)$$

where O is the output of neuron k, win is the weight between neuron k and activation n, 2 is the activation value of the previous layer, b is biased for neuron k, and is the activation function, e.g. sigmoid or ReLU.

CNN integrates this process in a deep, layered architecture to automatically extract complex hierarchical features from imagery. This capability allows CNN to identify patterns and objects in imagery with high accuracy, which is invaluable in real-world applications such as facial recognition, object detection, and medical analysis

accurate data.

II. METHODOLOGY

The research methodology for Classification of Brain Tumors Using CNN Deep Learning Method is designed with a systematic approach and involves a series of structured stages to ensure effective analysis and accurate results.

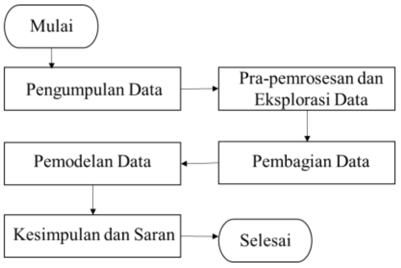
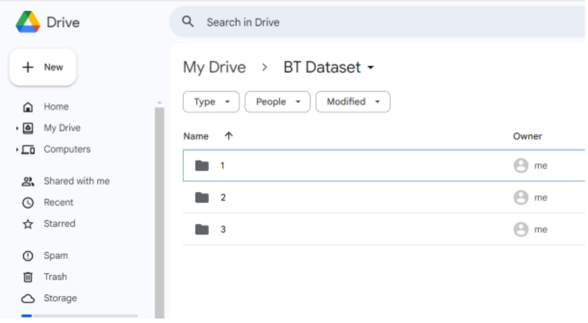


Figure 1. Flowchart Research

In accordance with the figure above, this research methodology has a series of systematic steps that begin with data collection, followed by pre-processing and exploration to prepare the data. After that, the data is divided into training and testing sets, which are then used to develop and test a brain tumor classification model using CNN. This process ends with the drawing of conclusions from the results obtained and the completion of the study. This process has the following details:

A. Data Collection

Before data exploration and pre-processing, data collection is carried out first because this stage is the first step in this project. The data used comes from a dataset of brain MRI images stored in Google Drive, covering various types of brain tumors such as gliomas, meningiomas, and pituitaries. Using Python libraries, this data is accessed and ready for further processing. Setting the location of the dataset folder and loading images of each tumor category is the first step to prepare the dataset for further analysis.



Name	Owner
1	me
2	me
3	me

Figure 2 Google Drive Data set

B. Data Pre-processing and Exploration

Once the data is collected, exploration and pre-processing of the data are carried out simultaneously to prepare the dataset for modeling. Data exploration involves the use of load_images_from_folder functions, which read images from different folders based on tumor type, collect images as well as associated labels, and visualize data distribution by plotting the number of images for each tumor category.

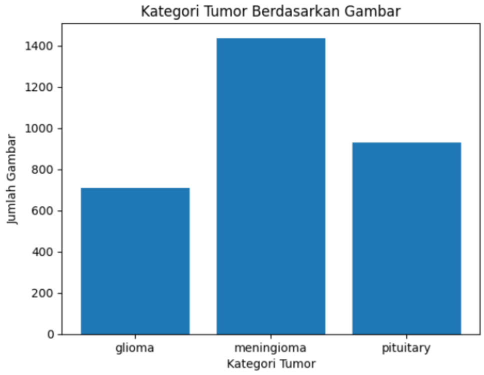


Figure 3. Barchart visualization of tumor categories based on images

Furthermore, the pre-processing step involves converting the image to grayscale to reduce the complexity and dimension of the data, resizing the image to a uniform size (128x128) for model input consistency, and applying Gaussian Blur and Median Blur filters to reduce noise [11]. This process uses OpenCV's cv2 library and normalizes pixel values by dividing by 255, converting data into ranges 0-1, a common practice in data preprocessing for deep learning models. Visualization of the comparison between the original image, after applying low-pass filtered, and median filtered provides insight into the effect of pre-processing on image quality.

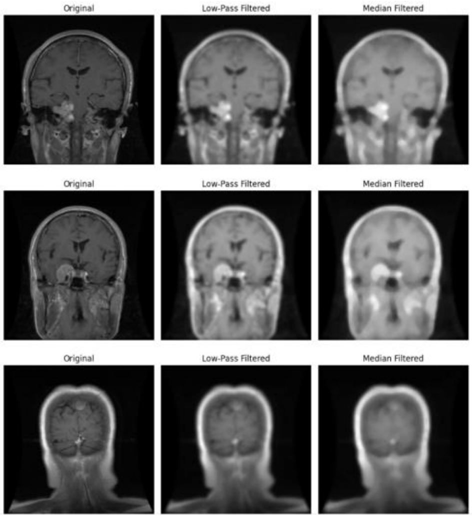


Figure 4. Visualization of Results from Brain Tumors

C. Feature Engineering

Image processing techniques play a crucial role in the diagnosis of brain tumors, turning medical images into rich sources of data for analysis. This process begins with image preprocessing, which includes techniques such as contrast enhancement to clarify image features, normalization to scale image intensity consistently, and noise reduction to reduce artifacts that could interfere with image interpretation [12]. Next, automatic feature extraction is performed to identify the most important attributes of the image, such as texture, shape, and intensity, which signal the presence of a tumor.

Image segmentation is an important next step, where algorithms separate tumor areas from normal brain tissue. Advanced segmentation techniques allow accurate detection of tumor boundaries, which is critical for determining tumor size, location, and characteristics [13]. The use of deep learning, particularly Convolutional Neural Networks (CNNs) has demonstrated remarkable capabilities in all aspects of this image processing, providing powerful tools to support clinical decisions with objective and accurate data.

After image processing, the next step is data classification, where deep learning models such as CNNs are used to categorize images based on the type of brain tumor presented. In this stage, the model is trained with a large dataset containing images that have been labeled according to the type of brain tumor, such as a meningioma, glioma, or pituitary tumor. During the training process, the model learns to recognize patterns and features unique to each tumor type, allowing it to accurately classify new images [14].

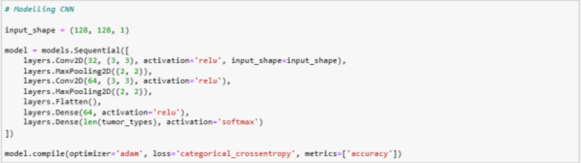
The use of CNN in the classification of brain tumors has shown significant improvements in diagnosis accuracy, surpassing traditional methods that rely on visual assessment by radiologists. These advances contribute greatly to the development of more effective treatment strategies, enabling early intervention and personalization of patient care based on the specific characteristics of their tumors [15].

D. Data Sharing

With the dataset processed, the next step is to split the dataset into training and test sets using the train_test_split function of the sklearn library. This division by the proportion of 80% for training and 20% for testing ensures that the model is trained on most data, and also has independent data sets for performance evaluation.

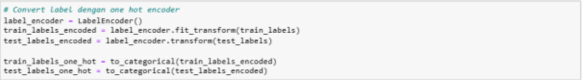
E. Modeling with CNN

Modeling is done using Convolutional Neural Network (CNN), which is a great choice for image processing due to its ability to extract image features hierarchically. The CNN model is built with multiple layers of convolution and pooling, followed by a dense layer for classification. This architecture is compiled with Adam's optimizer and loss function categorical_crossentropy to handle multi-class classification issues.



```
# Modelling CNN

input_shape = (128, 128, 1)

model = models.Sequential([
    layers.Conv2D(32, (3, 3), activation='relu', input_shape=input_shape),
    layers.MaxPooling2D((2, 2)),
    layers.Conv2D(64, (3, 3), activation='relu'),
    layers.MaxPooling2D((2, 2)),
    layers.Flatten(),
    layers.Dense(64, activation='relu'),
    layers.Dense(len(tumor_types), activation='softmax')
])

model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
```

Figure 5. CNN Coding Modelling



```
# Convert label dengan one hot encoder
label_encoder = LabelEncoder()
train_labels_encoded = label_encoder.fit_transform(train_labels)
test_labels_encoded = label_encoder.transform(test_labels)

train_labels_one_hot = to_categorical(train_labels_encoded)
test_labels_one_hot = to_categorical(test_labels_encoded)
```

Figure 6. Coding convert label with one hot encoder

F. Model Training and Evaluation

The model is trained with a training set using an accuracy matrix. The training process also involves validation with a subset of the training data to avoid overfitting. After training, the model is evaluated against a test set to assess its performance through a confusion matrix and classification report, which provides details about precision, recall, f1-score, and overall accuracy.

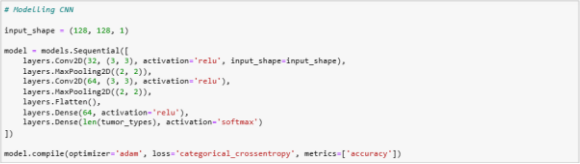
III. RESULTS AND DISCUSSION

A. Modeling Results

In this section, we will describe the results of modeling that has been carried out using the Convolutional Neural Network (CNN) method for brain tumor classification. The results of this modeling include analysis of model performance based on various evaluation matrices such as accuracy, precision, recall, and F1-score. The CNN model that has been built through an intensive training process is tested with validation data to ensure generalization and accuracy in the classification of three types of brain tumors: meningiomas, gliomas, and pituitary tumors.

The CNN model used in this study consists of several main layers, namely:

The CNN model used in this study consists of several main layers. The Convolution Layer is responsible for extracting features from the input image. Each convolution layer uses filters that move across the image, performing convolution operations to detect features such as edges, textures, and specific patterns. The Pooling Layer functions to reduce the dimensionality of the data generated from the convolution layer, thereby reducing complexity and the risk of overfitting. Max pooling is a frequently used method, where the maximum value of each feature patch is selected. The Fully Connected Layer integrates features that have been extracted from the previous layer to perform the final classification. Each neuron in this layer connects with all neurons in the previous layer, allowing the model to study complex combinations of features.

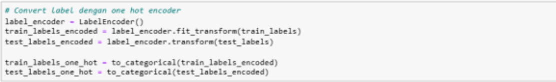


```
# Modelling CNN

input_shape = (128, 128, 1)

model = models.Sequential([
    layers.Conv2D(32, (3, 3), activation='relu', input_shape=input_shape),
    layers.MaxPooling2D((2, 2)),
    layers.Conv2D(64, (3, 3), activation='relu'),
    layers.MaxPooling2D((2, 2)),
    layers.Flatten(),
    layers.Dense(64, activation='relu'),
    layers.Dense(len(tumor_types), activation='softmax')
])

model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
```

Figure 1. CNN Modeling Coding

The CNN model's coding involves defining the image input form used in the model, with an input shape of 128x128 pixels in grayscale. A sequential model is created where layers are added one after another. The first convolution layer includes 32 filters of size 3x3 with a ReLU activation function and accepts the predefined image input shape. Following this is a max pooling layer with a pool size of 2x2 to reduce data dimensions. The second convolution layer has 64 filters of size 3x3 with a ReLU activation function, followed by another max pooling layer of the same pool size. The output of the last convolution layer is then flattened into a one-dimensional vector by the Flatten Layer. The first fully connected layer has 64 neurons with a ReLU activation function, and the final output layer has neurons corresponding to the number of tumor classes with a softmax activation function to perform the classification. The model is compiled using Adam's optimizer to update model weights during training, a categorical cross-entropy loss function suitable for multi-class classification problems, and accuracy metrics to evaluate model performance.

To convert labels using one hot encoder, an instance of LabelEncoder is created to convert category labels to numeric form. Training data labels are converted to numeric form using the fit_transform method, and the test data labels are transformed similarly. These numeric labels are then changed to one-hot encoded form for both the training and test data using the to_categorical method.

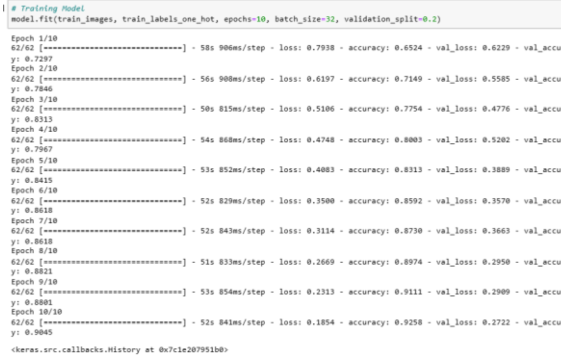


```
# Convert label dengan one hot encoder
label_encoder = LabelEncoder()
train_labels_encoded = label_encoder.fit_transform(train_labels)
test_labels_encoded = label_encoder.transform(test_labels)

train_labels_one_hot = to_categorical(train_labels_encoded)
test_labels_one_hot = to_categorical(test_labels_encoded)
```

Figure 2. Coding to convert labels with one hot encoder

To convert labels using a one-hot encoder, an instance of `LabelEncoder` is created to convert category labels to numeric form. The training data labels are converted to numeric form using the `fit_transform` method, while the test data labels are transformed similarly using the `transform` method. These numeric labels are then changed to one-hot encoded form for both the training and test data using the `to_categorical` method. This process ensures that the labels are in a format suitable for training the CNN model effectively.

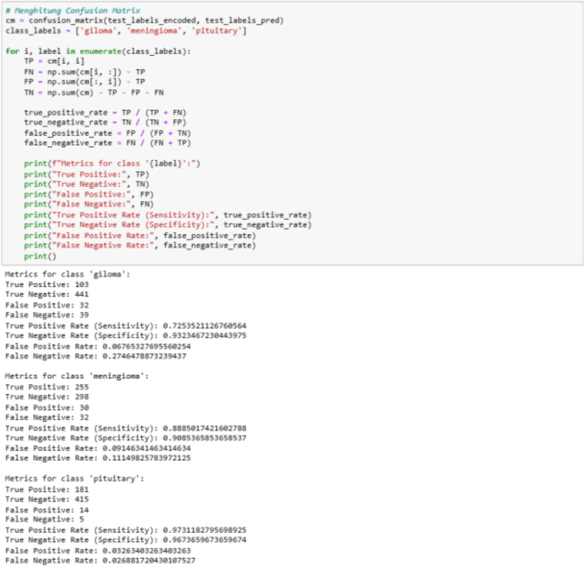
B. Results of Model Training and Evaluation



```
# Training Model
model.fit(train_images, train_labels_one_hot, epochs=10, batch_size=32, validation_split=0.2)
```

```
Epoch 1/10
62/62 [==============================] - 58s 906ms/step - loss: 0.7938 - accuracy: 0.6524 - val_loss: 0.6229 - val_accu
y: 0.7297
Epoch 2/10
62/62 [==============================] - 56s 908ms/step - loss: 0.6197 - accuracy: 0.7149 - val_loss: 0.5585 - val_accu
y: 0.7846
Epoch 3/10
62/62 [==============================] - 50s 815ms/step - loss: 0.5106 - accuracy: 0.7754 - val_loss: 0.4776 - val_accu
y: 0.8313
Epoch 4/10
62/62 [==============================] - 54s 868ms/step - loss: 0.4748 - accuracy: 0.8003 - val_loss: 0.5202 - val_accu
y: 0.7967
Epoch 5/10
62/62 [==============================] - 53s 852ms/step - loss: 0.4083 - accuracy: 0.8313 - val_loss: 0.3889 - val_accu
y: 0.8415
Epoch 6/10
62/62 [==============================] - 52s 829ms/step - loss: 0.3500 - accuracy: 0.8592 - val_loss: 0.3570 - val_accu
y: 0.8618
Epoch 7/10
62/62 [==============================] - 52s 843ms/step - loss: 0.3114 - accuracy: 0.8730 - val_loss: 0.3663 - val_accu
y: 0.8618
Epoch 8/10
62/62 [==============================] - 51s 833ms/step - loss: 0.2669 - accuracy: 0.8974 - val_loss: 0.2950 - val_accu
y: 0.8821
Epoch 9/10
62/62 [==============================] - 53s 854ms/step - loss: 0.2313 - accuracy: 0.9111 - val_loss: 0.2909 - val_accu
y: 0.8801
Epoch 10/10
62/62 [==============================] - 52s 841ms/step - loss: 0.1854 - accuracy: 0.9258 - val_loss: 0.2722 - val_accu
y: 0.9045

<keras.src.callbacks.History at 0x7c1e207951b0>
```

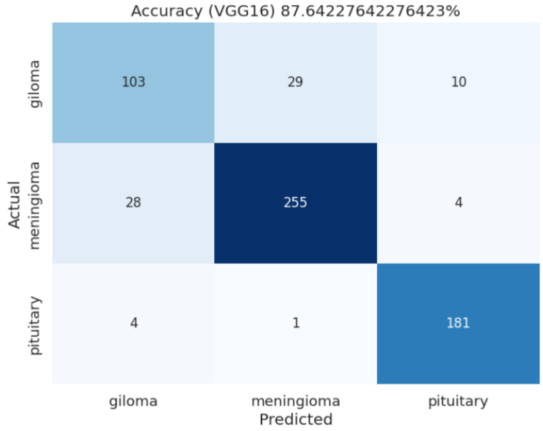
Figure 3.Training Model



```
# Menghitung Confusion Matrix
cm = confusion_matrix(test_labels_encoded, test_labels_pred)
class_labels = ['giloma', 'meningioma', 'pituitary']

for i, label in enumerate(class_labels):
    TP = cm[i, i]
    FN = np.sum(cm[i, :]) - TP
    FP = np.sum(cm[:, i]) - TP
    TN = np.sum(cm) - TP - FP - FN

    true_positive_rate = TP / (TP + FN)
    true_negative_rate = TN / (TN + FP)
    false_positive_rate = FP / (FP + TN)
    false_negative_rate = FN / (FN + TP)

    print(f"Metrics for class '{label}':")
    print("True Positive:", TP)
    print("True Negative:", TN)
    print("False Positive:", FP)
    print("False Negative:", FN)
    print("True Positive Rate (Sensitivity):", true_positive_rate)
    print("True Negative Rate (Specificity):", true_negative_rate)
    print("False Positive Rate:", false_positive_rate)
    print("False Negative Rate:", false_negative_rate)
    print()
```

```
Metrics for class 'giloma':
True Positive: 103
True Negative: 441
False Positive: 32
False Negative: 39
True Positive Rate (Sensitivity): 0.7253521126760564
True Negative Rate (Specificity): 0.9323467230443975
False Positive Rate: 0.06765327695560254
False Negative Rate: 0.2746478873239437

Metrics for class 'meningioma':
True Positive: 255
True Negative: 298
False Positive: 30
False Negative: 32
True Positive Rate (Sensitivity): 0.8885017421602788
True Negative Rate (Specificity): 0.9085365853658537
False Positive Rate: 0.09146341463414634
False Negative Rate: 0.11149825783972125

Metrics for class 'pituitary':
True Positive: 181
True Negative: 415
False Positive: 14
False Negative: 5
True Positive Rate (Sensitivity): 0.9731182795698925
True Negative Rate (Specificity): 0.9673659673659674
False Positive Rate: 0.03263403263403263
False Negative Rate: 0.026881720430107527
```

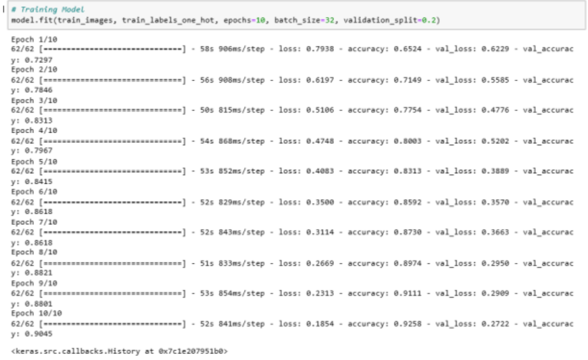
Figure 4 Confusion Matrix



```
              precision    recall  f1-score   support

      giloma       0.76      0.73      0.74       142
  meningioma       0.89      0.89      0.89       287
   pituitary       0.93      0.97      0.95       186

    accuracy                           0.88       615
   macro avg       0.86      0.86      0.86       615
weighted avg       0.87      0.88      0.88       615
```

Figure 5 Heatmap Visualization

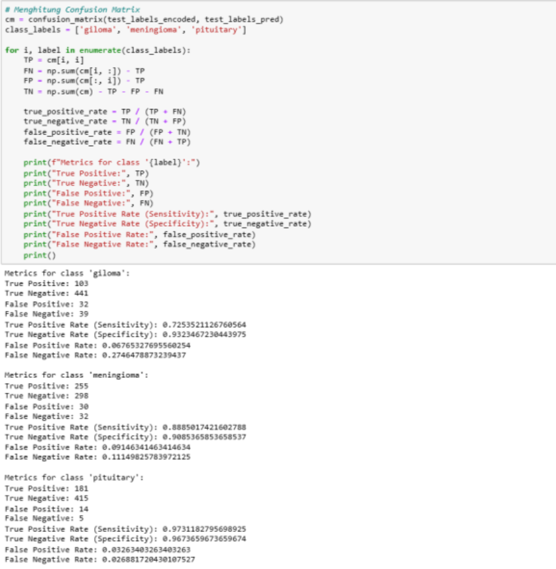
C. Discussion of Results Obtained



```
# Training Model
model.fit(train_images, train_labels_one_hot, epochs=10, batch_size=32, validation_split=0.2)
```

```
Epoch 1/10
62/62 [==============================] - 58s 906ms/step - loss: 0.7938 - accuracy: 0.6524 - val_loss: 0.6229 - val_accurac
y: 0.7297
Epoch 2/10
62/62 [==============================] - 56s 908ms/step - loss: 0.6197 - accuracy: 0.7149 - val_loss: 0.5585 - val_accurac
y: 0.7846
Epoch 3/10
62/62 [==============================] - 50s 815ms/step - loss: 0.5106 - accuracy: 0.7754 - val_loss: 0.4776 - val_accurac
y: 0.8313
Epoch 4/10
62/62 [==============================] - 54s 868ms/step - loss: 0.4748 - accuracy: 0.8003 - val_loss: 0.5202 - val_accurac
y: 0.7967
Epoch 5/10
62/62 [==============================] - 53s 852ms/step - loss: 0.4083 - accuracy: 0.8313 - val_loss: 0.3889 - val_accurac
y: 0.8415
Epoch 6/10
62/62 [==============================] - 52s 829ms/step - loss: 0.3500 - accuracy: 0.8592 - val_loss: 0.3570 - val_accurac
y: 0.8618
Epoch 7/10
62/62 [==============================] - 52s 843ms/step - loss: 0.3114 - accuracy: 0.8730 - val_loss: 0.3663 - val_accurac
y: 0.8618
Epoch 8/10
62/62 [==============================] - 51s 833ms/step - loss: 0.2669 - accuracy: 0.8974 - val_loss: 0.2950 - val_accurac
y: 0.8821
Epoch 9/10
62/62 [==============================] - 53s 854ms/step - loss: 0.2313 - accuracy: 0.9111 - val_loss: 0.2909 - val_accurac
y: 0.8801
Epoch 10/10
62/62 [==============================] - 52s 841ms/step - loss: 0.1854 - accuracy: 0.9258 - val_loss: 0.2722 - val_accurac
y: 0.9045

<keras.src.callbacks.History at 0x7c1e207951b0>
```

Figure 6. Training Model

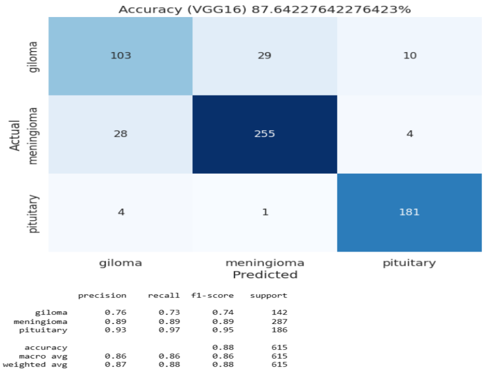


```
# Menghitung Confusion Matrix
cm = confusion_matrix(test_labels_encoded, test_labels_pred)
class_labels = ['giloma', 'meningioma', 'pituitary']

for i, label in enumerate(class_labels):
    TP = cm[i, i]
    FN = np.sum(cm[i, :]) - TP
    FP = np.sum(cm[:, i]) - TP
    TN = np.sum(cm) - TP - FP - FN

    true_positive_rate = TP / (TP + FN)
    true_negative_rate = TN / (TN + FP)
    false_positive_rate = FP / (FP + TN)
    false_negative_rate = FN / (FN + TP)

    print(f"Metrics for class '{label}':")
    print("True Positive:", TP)
    print("True Negative:", TN)
    print("False Positive:", FP)
    print("False Negative:", FN)
    print("True Positive Rate (Sensitivity):", true_positive_rate)
    print("True Negative Rate (Specificity):", true_negative_rate)
    print("False Positive Rate:", false_positive_rate)
    print("False Negative Rate:", false_negative_rate)
    print()
```

```
Metrics for class 'giloma':
True Positive: 103
True Negative: 441
False Positive: 32
False Negative: 39
True Positive Rate (Sensitivity): 0.7253521126760564
True Negative Rate (Specificity): 0.9323467230443975
False Positive Rate: 0.06765327695560254
False Negative Rate: 0.2746478873239437

Metrics for class 'meningioma':
True Positive: 255
True Negative: 298
False Positive: 30
False Negative: 32
True Positive Rate (Sensitivity): 0.8885017421602788
True Negative Rate (Specificity): 0.9085365853658537
False Positive Rate: 0.09146341463414634
False Negative Rate: 0.11149825783972125

Metrics for class 'pituitary':
True Positive: 181
True Negative: 415
False Positive: 14
False Negative: 5
True Positive Rate (Sensitivity): 0.9731182795698925
True Negative Rate (Specificity): 0.9673659673659674
False Positive Rate: 0.03263403263403263
False Negative Rate: 0.026881720430107527
```

Figure 7. Confusion Matrix

The confusion matrix for brain tumor classification using CNN in Deep Learning in this study showed a deep matrix for three types of tumors: glioma, meningioma, and pituitary. For gliomas, the model showed a true positive rate (sensitivity) of 0.7253 and a true negative rate (specificity) of 0.9323, indicating its strong ability to correctly identify these tumor types while also recognizing non-glioma cases appropriately. The meningioma classification achieved higher sensitivity and specificity, with levels of 0.8885 and 0.9085, respectively, reflecting the robustness of the model in distinguishing these tumor types. The model is best at classifying pituitary tumors, with an impressive sensitivity of 0.9731 and specificity of 0.9673, confirming remarkable precision for this category. This matrix shows that CNN models in Deep Learning can distinguish between different types of brain tumors effectively, offering great potential for improving early diagnosis and treatment planning.



```
              precision    recall  f1-score   support

      giloma       0.76      0.73      0.74       142
  meningioma       0.89      0.89      0.89       287
   pituitary       0.93      0.97      0.95       186

    accuracy                           0.88       615
   macro avg       0.86      0.86      0.86       615
weighted avg       0.87      0.88      0.88       615
```

Figure 8. Heatmap Visualization

The confusion matrix heatmap describes the performance of the Convolutional Neural Network (CNN) classification model in identifying three types of brain tumors: glioma, meningioma, and pituitary. From the visualization, we can see that the model was very accurate in classifying meningiomas with 255 correct predictions from 287 cases and had an impressive performance in detecting pituitary tumors with 181 correct predictions from 186 cases. Although the model was good at recognizing gliomas with 103 correct predictions out of 142 cases, there were a number of cases of gliomas that were misclassified as other types of tumors. Overall, this heatmap shows a model accuracy of 87.64227642276423%, indicating the effectiveness of the model in distinguishing between these types of tumors. The classification report under the heatmap confirms high levels of precision, recall, and f1-score for each category, with macro averages showing strong performance consistency across all tumor classes [16].

IV. CONCLUSIONS AND SUGGESTIONS

A. Conclusion

Based on the results of the study Classification of Brain Tumors Using the CNN Deep Learning Method, it can be concluded that the deep learning approach, especially the use of CNN, has high effectiveness in classifying types of brain tumors. By integrating image processing techniques and automated feature extraction, the developed model was able to produce a significant degree of accuracy in identifying gliomas, meningiomas, and pituitary tumors [17]. The confusion matrix heatmap provides a clear visualization of the model's performance, demonstrating the model's ability to recognize and distinguish between all three types of brain tumors with high precision, supported by an impressive sensitivity and specificity matrix.

Overall, this study provides strong evidence that deep learning technology, especially CNN, can be relied upon as an aid in the process of diagnosing brain tumors. These results hold promise in supporting radiologists and medical professionals, enabling more accurate diagnosis and the development of more personalized treatment plans based on the specific characteristics of each tumor type. With these technological advancements, the potential for significant improvements in the quality of care and treatment of brain tumor patients is becoming very open [18]. Further research with larger and more diverse datasets, as well as the development of more sophisticated model architectures, could pave the way for further improvements in accuracy and efficiency levels in brain tumor classification.

B. Suggestion

The suggestion for future research is to continue the research with larger and more diverse datasets. By using broader datasets, research can reinforce the generalizability of models and improve their accuracy. The use of diverse datasets is also important to ensure models can handle variations in medical images that may be encountered in clinical practice. In addition, the development of more sophisticated model architectures is also suggested to improve efficiency and accuracy in the classification of brain tumors.

REFERENCES

[1] K. Amalia, R. Magladena, and S. Saidah, "Classification of brain tumor disease on MRI images using CNN method with Alexnet architecture," 2022. https://openlibrarypublications.telkomuniversity.ac.id/index.php/engineering/article/view/18991

[2] L. O. Lukmana, "Brain tumor detection using CNN," fikom-unasman.ac.id, Oct. 2023, doi: 10.35329/jiik.v9i2.276.

[3]G. Litjens et al., "A survey on deep learning in medical image analysis," in Medical Image Analysis, vol. 42, pp. 60-88, Dec. 2017. [Online]. Available: https://doi.org/10.1016/j.media.2017.07.005

[4]A. Azhar, B. Siswoyo, D. Pratama, K. Anam, and H. Susana, "APPLICATION OF CONVOLUTIONAL NEURAL NETWORK (CNN) ALGORITHM FOR BRAIN TUMOR DIAGNOSIS," JATI, VOL. 8, NO. 2, PP. 1797–1801, Apr. 2024, doi: 10.36040/jati.v8i2.8242.

[5] D. Shen, G. Wu, and H.-I. Suk, "Deep Learning in Medical Image Analysis," in Annual Review of Biomedical Engineering, vol. 19, pp. 221-248, Jul. 2019. [Online]. Available: https://doi.org/10.1146/annurev-bioeng-071516-044442

[6]"IMPLEMENTATION OF CONVOLUTIONAL NEURAL NETWORK (CNN) METHOD IN DETECTING WASTE TYPE - UNJA Repository." https://repository.unja.ac.id/60591/

[7]D. Gunawan and H. Setiawan, "Convolutional Neural Network in Medical Imagery," Konconstellation, vol. 2, no. 2, May 2022, doi: 10.24002/konconstellation.v2i2.5367.

[8]P. H. Trenggono and A. Bachtiar, "THE ROLE OF ARTIFICIAL INTELLIGENCE IN HEALTH SERVICES: A SYSTEMATIC REVIEW," Journal of Ners, vol. 7, no. 1, pp. 444–451, Apr. 2023, doi: 10.31004/jn.v7i1.13612.

[9]E. H. Rachmawanto, D. Hermanto, Z. Pratama, and C. A. Sari, "PERFORMANCE OF CONVOLUTIONAL NEURAL NETWORK IN DEEP LAYERS RESNET-50 FOR MRI CLASSIFICATION OF BRAIN TUMORS," SEMNAS RISTEK (National Seminar on Research and Technology Innovation), vol. 8, no. 01, Jan. 2024, doi: 10.30998/semnasristek.v8i01.7125.

[10]N. Nafi'iyah, "Brain Tumor Identification MRI Image with Convolutional Neural Network," Journal of Informatics/Journal of Informatics: Journal of IT Development, vol. 8, no. 3, pp. 213–219, Sep. 2023, doi: 10.30591/jpit.v8i3.4985.

[11] M. R. Khilmawan and A. A. Riadi, "IMPLEMENTATION OF NOISE REDUCTION IN BONE IMAGES USING MEDIAN FILTER AND GAUSSIAN FILTER METHODS," JIPI (Scientific Journal of Informatics Research and Learning), vol. 3, no. 2, Dec. 2018, doi: 10.29100/jipi.v3i2.865.

[12] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," in Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015, 2015, doi: 10.1007/978-3-319-24574-4_28.

[13] A. Esteva et al., "Dermatologist-level classification of skin cancer with deep neural networks," Nature, vol. 542, pp. 115–118, 2017, doi: 10.1038/nature21056.

[14] H. R. Roth et al., "A new 2.5D representation for lymph node detection using random sets of deep convolutional neural network observations," in Medical Image Computing and Computer-Assisted Intervention – MICCAI 2014, 2014, doi: 10.1007/978-3-319-10470-6_58.

[15] K. He et al., "Deep Residual Learning for Image Recognition," in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, doi: 10.1109/CVPR.2016.90.

[16] N. Pratama, M. Liebenlito, and Y. Irene, "Comparison of Transfer Learning Convolutional Neural Network Classification Model for Brain Tumors using Magnetic Resonance Imaging Imagery," Jurnal Sehat Indonesia (JUSINDO), vol. 6, no. 01, pp. 308–318, Jan. 2024, doi: 10.59141/jsi.v6i01.81.

[17] M. Alizal, M. Susanto, A. Setyawan, H. Fitriawan, and M. Mardiana, "ROOM SECURITY SYSTEM WITH HUMAN DETECTION USING DEEP LEARNING-BASED CAMERA SENSORS," Alizal | Journal of Technoinfo, Jan. 2024, doi: 10.33365/jti.v18i1.3798.

[18] Z. M. Ilawanda and G. F. Atsani, "Radiological Features in the Neurological Field of Brain Tumors," Journal of Fusion Syntax, vol. 1, no. 12, pp. 987–1001, Dec. 2021, doi: 10.54543/fusion.v1i12.125.