Received: June 20, 2024. Revised: July 28, 2024.

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Multi-Model of Convolutional Neural Networks for Brain Tumor Classification in Magnetic Resonance Imaging Images

Irwan Budi Santoso[1]* **Supriyono[1]** **Shoffin Nahwa Utama[1]**

[1]*Informatics Engineering, Faculty of Science and Technology,*
Universitas Islam Negeri Maulana Malik Ibrahim, Malang, Indonesia
* Corresponding author's Email: irwan@ti.uin-malang.ac.id

Abstract: Classification of brain tumors based on magnetic resonance imaging (MRI) images is often carried out using convolutional neural network (CNN). However, the classification performance still needs to be improved due to the varying sizes, shapes, and positions of tumors and complex brain structures. In this study, we proposed a multi-model of CNN for brain tumor classification based on brain MRI images. The multi-model of CNN involves several CNN models (Xception, DensNet-201, and EfficientNet-B3), which were constructed using the proposed algorithm. This algorithm works by combining the advantages of each CNN model using classification results rules, which are formed based on the highest and smallest accuracy and false positive values from training validation. The first model in the multi-model structure can be selected from the CNN model with the smallest or largest validation accuracy and connected to the CNN model with the lowest false positives. We used brain tumor MRI image datasets to evaluate the algorithm's performance, including the THOMAS dataset (Dataset 1) and the NICKPARVAR dataset (Dataset 2). The test results showed that the multi-model of CNN constructed with this algorithm produces the best accuracy of 97.74% for Dataset 1 and 99.69% for Dataset 2. From these results, the multi-model of CNN can outperform the single CNN model with an accuracy improvement of 1.29%-4.19% for Dataset 1 and 0.22%-0.61% for Dataset 2.

Keywords: Magnetic resonance imaging, Brain tumor, False positive, Validation, Convolutional neural network.

1. Introduction

In supporting diagnosis, radiologists can classify the type of brain tumor by examining the MRI scan results [1]. MRI images from these scans can provide better spatial information or visualization [2, 3]. Classification of brain tumors based on MRI images is essential for determining the next medical steps. However, manually classifying the type of brain tumor from the imaging results can take much time and has the potential for errors. The development of automatic methods is currently one solution that can help overcome this problem.

One deep learning method widely used for classifying tumor types based on MRI images is CNN. In previous research, many proposed methods involved one or more CNN models for MRI image feature extraction, while at the classification stage using conventional machine learning, as has been reported in [4-6]. The results of the test they conducted showed that CNN contributed to improving tumor classification performance, even though it was only for feature extraction. Efforts to improve classification performance are also made using the full CNN model, not just for feature extraction. Their proposed preprocessing could improve the tumor classification performance of CNN models, as shown in [7, 8]. Combining CNN models through combining convolution processes, architectures, and features from several architecture blocks and combining network paths in one architecture also contributes to tumor classification performance [9-11]. Chatterjee et al. [9] proposed a ResNet Mixed Convolution model obtained by combining two 2D and 3D convolution processes of ResNet 2D and ResNet 3D. Khan et al. [10] combined the VGG16 architecture via transfer learning with a reflection 23-layer CNN architecture. Noreen et al. [11] combined multi-level features, first combining features extracted from various pre-trained Inception-V3 model modules and passing a softmax classifier. Second, they combined the features extracted from multiple pre-trained DensNet-201 blocks and passed them to a softmax classifier to classify brain tumors.

Each CNN model is designed with a different number of convolutions and convolution sizes, which can produce different tumor classification performances [12, 13]. CNN models with different structures may provide different advantages, and when combined, they will provide better classification performance. Therefore, combining models that have different convolution sizes or dimensions can improve classification performance [9, 10]. Combining features from several blocks in a CNN model can also improve tumor classification performance [11]. However, with the shape and size of brain tumors varying wildly, the position of the tumor in the brain was difficult to ascertain [14], and the complex structure of the brain that was only combining the model structure, the convolution, and the features as done was not sufficient to obtain the best classification performance. The best solution to improve tumor classification performance is to combine the advantages of several CNN models and leave behind their weaknesses.

The benefits of a model can be known clearly after looking at the results of the test confusion matrix, which cannot be seen during the learning process. The confusion matrix from the validation dataset during learning can be a prototype to see the advantages of one model over another based on the accuracy values and false positives produced. Therefore, this study proposes an algorithm for combining several CNN models based on the confusion matrix from the validation dataset to construct a multi-model of CNN, resulting in better tumor classification performance than a single CNN model. Transfer learning of several well-known CNN models, such as, Xception [15], DensNet-201 [16], and EfficientNet-B3 [17] can help provide better performance CNN multi-model. Xception, DensNet-201, and EfficientNet-B3 in implementing tumor classification were proven to have the best performance [11, 18, 19].

Based on this explanation, the main contributions of this study are as follows:

- We constructed the CNN multi-model involving Xception, DensNet-201, and EfficientNet-B3 transfer learning models.
- We proposed an algorithm to construct a multi-model of CNN involving several CNN models based on accuracy values and false positives on the validation dataset during the MRI image-based learning process.
- We conducted a comparative analysis of classification performance between the proposed CNN multi-model and the single CNN model.

In this paper, Section 2 discusses relevant previous research methods on brain tumor classification. Section 3 describes the MRI dataset to evaluate the proposed method and the steps of the method. Section 4 describes the parameters and scenarios of the tests carried out. Section 5 contains all experimental results and discussion. The final session, Section 6, includes the conclusion.

2. Related work

Previous studies reported the involvement of CNN models for feature extraction of brain MRI images, while other classifiers were used for brain tumor classification. Kang et al. [4] proposed a hybrid scheme involving thirteen transfer learning CNN models to extract MRI image features. They also involved nine machine learning classifiers to evaluate the features and perform the final classification. The test results showed that the combined feature extraction results from DenseNet-121, ResNeXt-101, and MnasNet with the final classification using fully connected provided the best performance. Ahmad & Choudhury [5] proposed a research scheme that was almost the same, namely using seven transfer learning CNN models for feature extraction of brain MRI images and involving five machine learning classifiers for classification. The evaluation results showed that the VGG19-SVM combination they proposed achieved better performance than other combinations, with an accuracy of 99.39%. Starting with pre-processing with a Gaussian filter, Shanthi et al. [6] also extracted brain MRI image features with CNN. They used long short-term memory (LSTM) with weights determined using the adaptive rider optimizer (ARO) algorithm at the classification stage. The evaluation results showed that their proposed method yielded a maximum accuracy of 97.5%.

Previous research also developed the use of the CNN model for feature extraction and tumor classification. Chatterjee et al. [9] proposed ResNet(2+1)D and Mix ResNet to classify brain tumor types. The ResNet(2+1)D model they proposed was designed by combining 2D convolution and 1D convolution, while the Mix ResNet combines 2D and 3D convolution. Their evaluation showed that the two proposed models produced better performance than ResNet3D. Their tests showed that the proposed model was superior to the ResNet3D model, with an accuracy of 96.98%.

In addition to convolutional combining, Khan et al.[10] proposed ensembling two CNN models, namely a 23-layer CNN designed with VGG16 transfer learning. They ensembled these models to overcome overfitting on small data sets. The test results showed that their proposed method achieved an accuracy of up to 97.8% for large and 100% for small datasets. Younis et al. [20] conducted research using almost the same method, namely ensembling the CNN and VGG16 models to overcome overfitting. The test results of the ensemble of two models obtained an accuracy of 98.14%.

Noreen et al. [11] carried out a different combining, namely fusing multi-level features of the CNN model. The first fusion is performed by fusing features extracted from the Inception-v3 module, and the second fusing features extracted from DensNet201 blocks. They used softmax for tumor classification by entering each feature fusion. The evaluation results of the first and second fusion produced accuracies of 99.34% and 99.51%, respectively. A different thing was conducted by Asif et al. [18], which did not fusion features from CNN blocks. They only used the transfer learning model Xception, NasNet Large, DenseNet121, and InceptionResNet-V2, and at each classification layer of the model, they used a softmax. Evaluation results showed that Xception yielded the best performance with an accuracy of 99.67% for large and 91.94% for small datasets.

Deep pre-processing of MRI images in several studies also improved tumor classification performance. Rizwan et al. [7] proposed a Gaussian convolutional neural network (GCNN) using Gaussian filters to improve tumor classification performance. The evaluation results of their proposed method, GCNN, obtained the best accuracy of 99.8% for the first dataset and 97.14% for the second dataset.

Musallam et al. [8] proposed three pre-processing steps and a deep convolutional neural network (DCNN) to classify tumor types. Cropping, denoising with a non-local mean algorithm, and histogram equalization were the three pre-processing steps that contributed to the performance of DCNN in tumor classification with an overall accuracy of 98.22%. Jun & Liyuan [21] demonstrated different steps in improving CNN performance by using a gated channel transformation (GCT) layer. Their proposed CNN involved parallel multipath networks, each with the same number and size of convolution filters. The trial results showed their proposed method achieved 98.61% accuracy in classifying tumor types.

A different study was conducted by Shilaskar et al. [22], which did not use CNN but had almost the same stages. Before the classification stage, they proposed a histogram of oriented gradients (HOG) for brain MRI image feature extraction. In the classification stage, they involved several machine learning, including support vector machine (SVM), gradient boost, k-nearest neighbor, extreme gradient boosting (XG Boost), and logistic regression. Test results showed that HOG with XG Boost yielded the highest accuracy of 92.02% compared to other classifiers. Atha & Chaki [23] proposed a semi-supervised brain tumor classification network (SSBTCNet) approach. Their proposed SSBTCNet combined unsupervised autoencoder (AE) with supervised classification networks. The semi-supervised aimed to assist in adapting hidden descriptor learning for classification purposes. The evaluation results of their proposed method obtained a classification accuracy of 96.5%.

Extraction of brain MRI image features using one or more CNN models, as shown in [4-6], is one of the best efforts to improve classifier performance. However, separating the extraction of brain MRI image features from a classifier does not guarantee that the resulting features are the representative of the the classifier use. The reality of brain MRI image data for input to the CNN model is two-dimensional (2D). The combination of 2D convolution with 1D or the combination of 2D and 3D reported in [9] is the same as only performing 2D convolution that impacts classification performance. This condition is almost the same as the multi-level feature fusion in one model shown in [11], basically the same as using the features produced by the original model. Likewise, the multipath network with the same filter size for each path in [21] is the same as using a path with the same filter size. On the other hand, an ensemble of two CNN models, which aims to avoid overfitting as in [10, 20], is one solution to cover the weaknesses of one of the other models. The work is the same as testing several CNN models and selecting the best performance among the CNN models, as shown in [18]. Likewise, pre-processing brain MRI images to remove noise and clarify the input image in [7, 8] can improve classification performance. However, the reality is that tumor types have very diverse sizes, shapes, and tumor positions, and using only a CNN model will also make it not sufficient to get the best performance in brain tumor classification.

One of the best options is involving several CNN models for classifying brain tumors with high shape, size, and position variations on MRI images. Therefore, we propose a multi-model of CNN for brain tumor classification based on brain MRI images in this study. The proposed method in this study is different from other methods in three main aspects;

they are: (i) constructing a multi-model of CNN, which includes three models, namely, Xception, DensNet-201, and EfficientNet-B3, (ii) the method scheme for building a multi-model of CNN involves several CNN models with input MRI images and several epochs in learning, (iii) the proposed algorithm for building a multi-model of CNN is based on the accuracy values and the false positive on validation, that is combining the advantages of the CNN model by looking at the smallest false positive value of each CNN model.

3. Material and methods

3.1 Dataset of experiment

To evaluate our proposed algorithm, we used two brain MRI image datasets. The first dataset (Dataset 1) is the THOMAS dataset, which contains brain MRI images for brain tumor classification and can be accessed on the Kaggle website [24]. The dataset refined the SARTAJ dataset [25]. The dataset consists of brain MRI images (axial, sagittal, coronal), which contain the types of glioma, meningioma, and pituitary tumors, as well as the no tumor. The dataset's total number of MRI images is 3096, consisting of 901 glioma MRI images, 913 meningioma MRI images, 844 pituitary MRI images, and 438 normal (no tumor) MRI images. The second dataset (Dataset 2) is the NICKPARVAR dataset, which contains brain MRI images (axial, sagittal, and coronal) and can be obtained from the Kaggle website [26]. The dataset is a combination of public datasets, including Figshare [27], SARTAJ [25], and BR35H [28]. The dataset has been separated into datasets for training and testing. The dataset for training includes 1321 MRI images of glioma, 1339 meningioma, 1457 pituitary, and 1595 no tumor. Thus, the total number of MRI images for training is 5712. Meanwhile, the dataset for testing includes glioma, meningioma, pituitary, and no tumor, with the respective number of brain MRI images 300, 306, 300, and 405, respectively. Accordingly, the total dataset for testing is 1311 brain MRI images. Examples of brain MRI images containing glioma, meningioma, pituitary, and no tumor are shown in Fig. 1.

In each evaluation of the CNN model and the proposed algorithm, we arranged the first and second datasets into training, validation, and testing datasets. For the first dataset, splitting was performed with a composition of 10% of the total dataset, namely 310 brain MRI images were the testing dataset. We split the remaining dataset again with a composition of 10%, namely 279 MRI images were the validation dataset, and the remaining 2507 brain MRI images were the training dataset. In the second dataset, there are 1311 brain MRI images for testing and 5712 brain MRI images for training. Next, we split the training dataset with a composition of 10% for validation. Therefore, there are 572 brain MRI images for validation and 5140 MRI images for the model training process. Details on splitting the dataset can be seen in Table 2.

Table 2. The composition of brain MRI images dataset for evaluation.

Dataset	Label	Tr	Val	Ts
THOMAS dataset (Dataset 1) [24]	glioma	730	81	90
	meningioma	740	82	91
	pituitary	683	76	85
	no tumor	354	40	44
	Total	2507	279	310
NICKPARVAR dataset (Dataset 2) [26]	glioma	1189	132	300
	meningioma	1205	134	306
	pituitary	1311	146	300
	no tumor	1435	160	405
	Total	5140	572	1311

*Tr=Training, Val=Validation, Ts=Testing

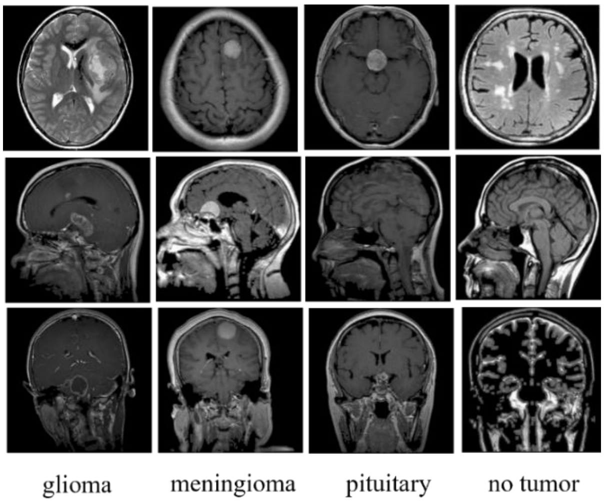


Figure. 1 Examples of brain MRI images: glioma, meningioma, pituitary, and no tumor

3.2 Transfer learning

Transfer learning is a deep learning technique that uses models trained on large datasets such as ImageNet [29] to initialize model training on different datasets. In this study, transfer learning helps train CNN models on small datasets, such as brain MRI image datasets [30]. The main advantage of transfer learning is that it prevents overfitting, thereby improving brain tumor classification performance. The transfer learning CNN models chosen in this study are Xception, DenseNet-201, and EfficientNet-B3, which have been proven in many studies to provide the best performance.

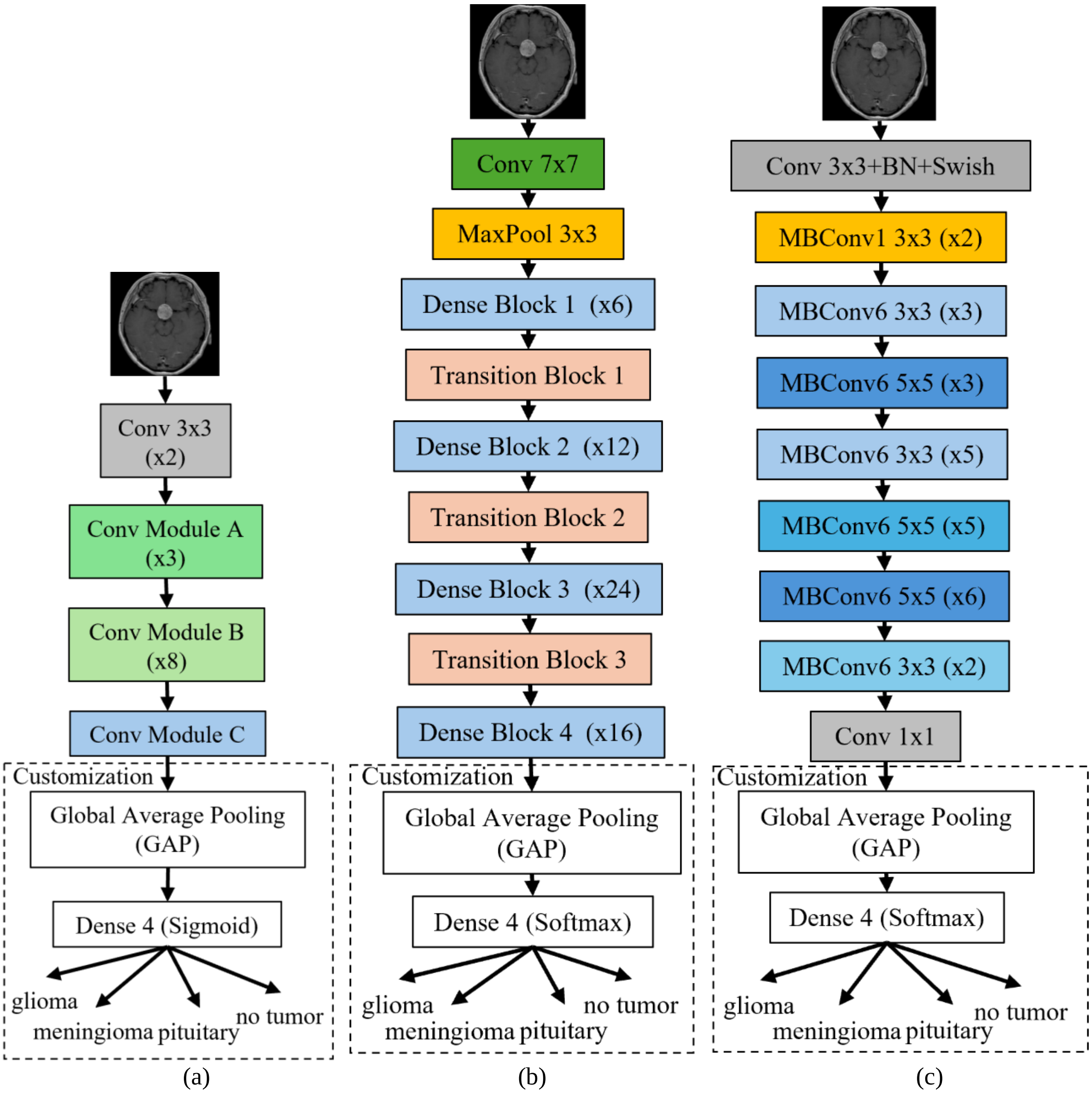


Figure. 2 Basic architecture and customization in top-level classification: (a) Xception, (b) DensNet-201, and (c) EfficientNet-B3

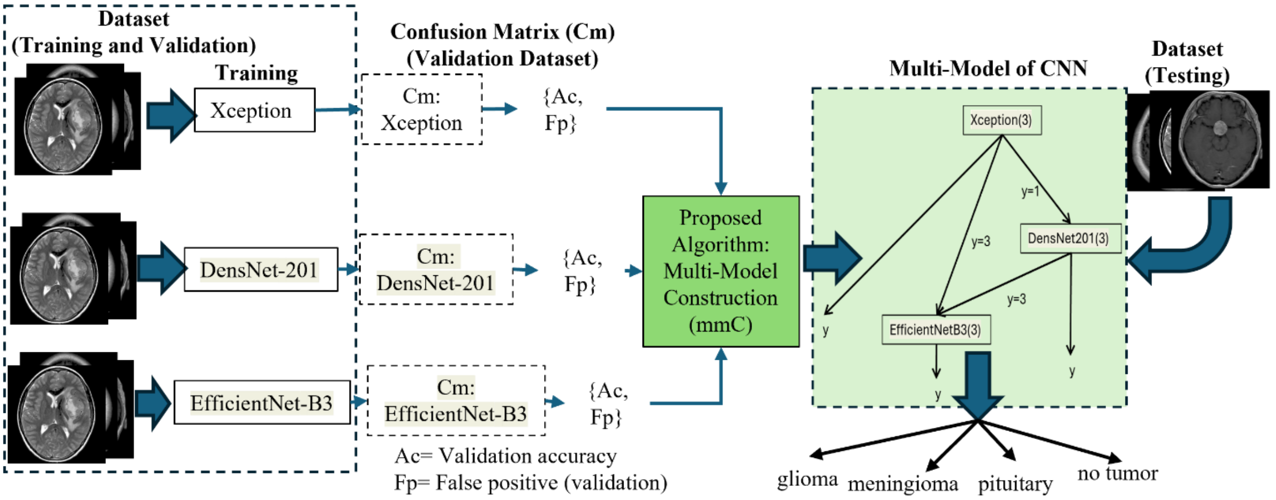


Figure. 3 The proposed scheme: tumor classification using multi-model of CNN

3.3 Xception

Xception is a CNN architecture that performs best as a continuation of the Inception architecture [15]. The architecture has 36 convolutional layers divided into 14 different modules. Each module other than the start and end modules has residual linear links around it. This architecture has a convolution operation with a filter size of 1 × 1 to get the cross-channel correlation. The correlation is considered a 2D + 1D mapping. In this study, we adjusted the top layer of the model architecture with a classification layer to classify brain tumors using an activation function. Fig. 2 (a) shows the basic architecture of the Xception and adjustments to the layers.

3.4 DensNet-201

DenseNet is a deep learning model that connects each layer to each subsequent layer using feedforward [16]. CNN models with L-layers generally have L connections, while DenseNet has direct connections of (L×(L+1))/2. Each layer in the model has a feature map as input to the next layer. To avoid overfitting, DenseNet applies regularization to the learning process. In this research, the model used is DenseNet-201, containing four dense blocks, each of which has 6, 12, 24, and 16 convolution blocks. In this research, we also adjusted the top layer of the model architecture with a classification layer for brain tumor classification cases. Fig. 2 (b) shows the basic architecture of the Densenet-201 adjusted for the classification layer.

3.5 EfficientNet-B3

EfficientNet is a deep learning model architecture that uses a scaling method to scale all dimensions uniformly using combined coefficients [17]. The CNN model architecture has several variants, and in this study, we used EfficientNet-B3. We used these variants due to considerations of computational resources and accuracy. The model architecture has two convolution layers, seven blocks of mobile bottleneck convolution (MBConv), one pooling layer, and a fully connected layer. At each layer in the MBConv block, an inverted residual connection is carried out except for the first layer in the block. With this activity, the model can train deeper neural networks to capture more prosperous and complex features, thereby improving performance. Fig. 2 (c) shows the original architecture of the EfficientNet-B3, which has been adjusted for the classification layers.

3.6 Customization CNN model

After the last convolution layer in the CNN model, all the features resulting from the convolution process are converted into a vector before the classification stage. Global Average Pooling (GAP) is a pooling operation other than the flatten operation that can be used to convert features into a vector and has been proven to be robust in practice [31-33]. All CNN models (Xception, DenseNet-201, and EfficientNet-B3) use this operation in this study. Mathematically, the GAP operation is shown in Eq. (1).

$$g_i(f_i) = \frac{1}{NN}\sum_{j=1}^{NN} f_{ij} \tag{1}$$

with f_i is the i^{th} feature map, g_i is the GAP output on f_i. f_{ij} is the feature vector contained in f_i, and NN is the number of elements in the feature vector.

The results of the GAP process for each feature map are then used as input to the fully connected layer and go directly to the classification layer. In the Xception model, adjustments are made to the classification layer with four neurons according to the number of labels/classes (glioma, meningioma, pituitary, and no tumor), with the activation function used being sigmoid (logistic). Mathematically, this function is shown in Eq. (2) [34].

$$\hat{h}_k(\hat{g}_k) = \frac{1}{1+\exp(-\hat{g}_k)} \tag{2}$$

where $\hat{h}_k$ is the sigmoid value in the k^{th} class, and $\hat{g}_k$ is the result of the fully connected layer process in the k^{th} class. Meanwhile, the DenseNet-201 and EfficientNet-B3 models use the softmax activation function, as shown in Eq. (3) [35, 36].

$$\tilde{h}_k(\hat{g}_k) = \frac{\exp(\hat{g}_k)}{\sum_{j=1}^{C}\exp(\hat{g}_j)} \tag{3}$$

with $\tilde{h}_k$ as the softmax value in the k^{th} class, and C is the number of classes (there are 4, namely glioma, meningioma, pituitary, and no tumor). The selection of activation functions in each model is based on the recommendations of the original model. Meanwhile, the loss function used by each CNN model during the optimization process in this research is categorical cross-entropy [19,32]. Categorical cross-entropy mathematically in this study is shown in Eq. (4).

$$\mathcal{L}(z_k, \hat{h}_k) = -\frac{1}{S}\sum_{i=1}^{S}\sum_{k=1}^{C} z_{ik}\log \hat{h}_{ik} \tag{4}$$

with S as the number of brain MRI images in batch, z_k is the actual k^{th} class value, and $\hat{h}_k$ is the predicted value of the k^{th} class (for DenseNet-201 and EfficientNet-B3, $\hat{h}_k$ replaced $\tilde{h}_k$).

3.7 Proposed algorithm multi-model of CNN

To improve the performance of the models selected during testing (Xception, DenseNet-201, and EfficientNet-B3), in this study, we proposed an algorithm to construct a multi-model CNN called mmC. This algorithm works by combining the advantages of the classification results of each CNN model based on the validation dataset confusion matrix. The initial step before implementing the algorithm is training each CNN model (a single CNN model) to get the model weights and validation confusion matrix on the last epoch. The confusion matrix for each model becomes the proposed algorithm's input, as shown in Fig. 3.

Table 1. Notation list

Notation	Description
g	Global average pooling output
f	Feature map
z	Actual label value
$\hat{h}$	Sigmoid output
$\hat{g}$	Fully connected layer output
$\tilde{h}$	Softmax output
$\mathcal{L}$	Categorical cross-entropy output
Cm	Validation confusion matrix
M	CNN model
Tp	True positive
Fp	False positive
Tn	True negative
Fn	False negative
Ac	Accuracy of validation
$fnode$	Option of the first CNN model in the structure of multi-model
$minFp$	Minimum of Fp in each label
$maxFp$	Maximum of Fp in each label
$IdxM$	CNN model index
$G(V,E)$	Graph structure of CNN multi-model
V	Set of models in CNN multi-model
E	Set of label value for rule in CNN multi-model
$LabelSort$	Sort of $maxFp$ in descending
lb	Label value for rule in CNN multi-model
h	Prediction using parameter of CNN
y	Label prediction (classification result)
Acr	Accuracy of testing
Prs	Precision of testing
Svt	Sensitivity of testing
Sft	Specificity of testing
$F\text{-}sc$	F-score of testing

We have prepared a notation list to help understand the notations in CNN models, the proposed algorithm, and the scheme, as shown in Table 1.

Algorithm 1 Multi-Model Construction (mmC)

Input: Cm_1 , Cm_2 ,..., Cm_R *{confusion matrix of validation, 1st model (M_1), 2nd model (M_2), ..., Rth model (M_R)}*, $fnode$ *{0 (min Ac), 1 (max Ac)}*, D *{Total of Image Validation}*, K *{Number of labels}*

Output: $G(V, E)$ *{V=set of sequential models in mmC, E=set of sequential values (labels) for rule in mmC}*

{True positive = Tp , False positive = Fp}

```
for $i$←0 to $K$-1
    for $j$←1 to $R$
        $Tp_{ij}$ ←countTp($Cm_j$)
        $Fp_{ij}$ ←countFp($Cm_j$)
for $j$←1 to $R$ {number of models}
    $Ac_j$←sum($Tp_j$)/$D$
{Validation accuracy of each model}
$V$←{ }
if $fnode$=0
    $IdxM$← $\operatorname{argmin}_{1≤i≤R}(Ac)$
    $V$← $V \cup M_{IdxM}$ {first model in mmC}
else
    $IdxM$← $\operatorname{argmax}_{1≤i≤R}(Ac)$
    $V$← $V \cup M_{idxM}$ {first model in mmC}
    {If min (max) accuracy more than a model, choose one}
for $i$←0 to $K$-1
    $minFp_i$ ← min ($Fp_i$) {Min $Fp$}
    $maxFp_i$ ← max ($Fp_i$) {Max $Fp$}
$LabelSort$ ← sort ($maxFp$) {Sort $maxFp$ : descending}
$E$←{ }
for $j$←1 to $K$ {can use some labels}
    ($lb_j$, $IdxM_j$)←Seach($LabelSort_j$, $minFp$)
        {if model in a label more than a model, choose one}
    $V$←$V \cup M_{IdxM_j}$
    $E$←$E \cup lb_j$
return $G(V,E)$
```

Algorithm 2 Testing of mmC (4 models)

Input: $G(V, E)$

{ V= {M_1,M_2,M_3,M_4}, E={lb_1,lb_2,lb_3} , X *{testing MRI image}*, K *{Number of labels}*

Output: y *{result of classification}*

```
$h_1$←$M_1(X)$ {prediction using parameter of $M_1$}
```

```
2    $y \leftarrow \underset{0 \le i \le K-1}{\operatorname{argmax}}(h_1)$ {label prediction using $M_1$}
3    if $y = lb_1$ {rule for model connection}
4        $h_2 \leftarrow M_2(X)$ {prediction using parameter of $M_2$}
5        $y \leftarrow \underset{0 \le i \le K-1}{\operatorname{argmax}}(h_2)$ {label prediction using $M_2$}
6    if $y = lb_2$
7        $h_3 \leftarrow M_3(X)$ ) {prediction using parameter of $M_3$}
8        $y \leftarrow \underset{0 \le i \le K-1}{\operatorname{argmax}}(h_3)$ {label prediction using $M_3$}
9    if $y = lb_3$
10       $h_4 \leftarrow M_4(X)$ ) {prediction using parameter of $M_4$}
11       $y \leftarrow \underset{0 \le i \le K-1}{\operatorname{argmax}}(h_4)$ {label prediction using $M_4$}
12 return $y$
```

Algorithm 1 is an algorithm proposed to form a multi-model of CNN involving several CNN models**.** The algorithm input includes the validation confusion matrix for each CNN model to be combined, the first CNN model in the multi-model (*fnode*), the total image for validation (*D*), and the number of labels (*K*). The validation confusion matrix produced by the j^{th} CNN model is shown by Cm_j, where j = 1,..., *R*, and *R* are the numbers of CNN models to be combined. The first step in the algorithm is to determine the true positives (Tp_{ij}) and the false positives (Fp_{ij}) for each label and CNN model that uses the countTp function. Tp_{ij} is the true positive of validation on the i^{th} label and j^{th} CNN model, while Fp_{ij} is the false positive of validation on the i^{th} label and j^{th} CNN model. The next step is determining validation accuracy for each CNN model (Ac_j) using Algoritma 1 line 6 with j = 1,..., *R*. The sum function adds up all *Tp* for each label in Cm_j produced by the j^{th} CNN model. In this study, there are four labels, namely 0, 1, 2, and 3, which respectively indicate glioma tumor, meningioma tumor, no tumor, and pituitary tumor. Furthermore, determine the first CNN model that enters the multi-model CNN structure. If *fnode* = 0, the first model to enter the multi-model structure is the model with the lowest validation accuracy. Selecting the first model begins with determining the model indeks (*IdxM*) that has the lowest validation accuracy with the argmin function for each validation accuracy (*Ac*) produced by the CNN model (see Algorithm 1, line 9). Meanwhile, if *fnode*=1, the first model to enter the multi-model structure is the model with the highest validation accuracy. *IdxM* in this condition is the model index with the highest model accuracy obtained using the argmax function (see Algorithm 1, line 12). M_{IdxM} represents the first incoming CNN model and is then collected into the set *V* with the operation $V \cup M_{IdxM}$ ={} ∪ M_{IdxM} = {M_{IdxM}} (see Algorithm 1, line 10 or 13). As a note, if more than one model has the lowest or highest accuracy, choose one. The next step is to determine the next CNN model that enters the multi-model CNN structure. The next step is determining the smallest and largest false positive values for each label based on Fp_i produced by the CNN model and indicated by $minFp_i$ and $maxFp_i$ (see Algorithm 1, lines 15 and 16). The min and max functions in the algorithm are to get the minimum and maximum values of the false positive of validation produced by each model. In the next step, sort the maximum false positive value for each label ($maxFp_i$) in descending order using the sort function, and the results are stored in *LabelSort* (see Algorithm 1, line 17). The next step is to determine the CNN model index ($IdxM_1$) that enters the following multi-model structure and the label value (lb_1) based on the smallest false positive value (*minFp*) on the label that contains the highest $LabelSort_j$ (j=1) value using the Search function (see Algorithm 1, line 20). The label value (lb_1) is a linking rule with other models in a multi-model structure. The same is done at the next highest $LabelSort_j$ (j=2,…, *K*) value to determine the model index and next label value. The model with the selected index at j=1 is then added to the set *V* with the operation $V \cup M_{IdxM_j}$ ={M_{IdxM}} ∪ M_{IdxM_j} = {M_{IdxM}, M_{IdxM_1}} and labels value enter into the set *E* with the operation $E \cup lb_j$ = {} ∪ lb_j = {lb_1} (see Algorithm 1, lines 21 and 22). In the same way, it is conducted for j=2,…, *K*. As a note, if there is more than one model with the lowest false positive, then just one model is selected. The algorithm's output is *G*(*V*, *E*), where *V* is a set of CNN models selected sequentially in the multi-model of CNN, and *E* is a set of label values for the rules that connect the CNN models in it. In testing, *G*(*V*, *E*) will be implemented as a combination of the classification results of CNN models based on the sequence of models and rules. An example of implementing the multi-model of CNN in testing involving four CNN models can be seen in Algorithm 2.

The input of Algorithm 2 is the model structure resulting from the construction of a multi-model CNN with mmC (*G*(*V*, *E*)) and the brain MRI image for testing (*X*). *G*(*V*, *E*) is a multi-model structure with *V*={M_1, M_2, M_3, M_4}and *E*={lb_1, lb_2, lb_3}. M_1, M_2, M_3, and M_4 are the first, second, third, and fourth CNN models in the multi-model structure of CNN. In contrast, lb_1, lb_2, and lb_3 in the set *E* are the label values as rules connecting the second, third, and fourth CNN models in the multi-model CNN structure. The first step of Algorithm 2 predicts the softmax/sigmoid value on the output layer (h_1) of the first CNN model (M_1) based on the parameters of the model's learning results using input brain MRI image (X) using Eqs. (2) or (3). Moreover, predict the label (y) with the input value h_1 using the argmax function (See Algorithm 2, line 2). The prediction results enter the rule condition if $y = lb_1$. If $y = lb_1$, then the value of y is replaced with the value of y obtained from label prediction with input h_2, which is predicted using the second CNN model (M_2). Next, y obtained from h_1 and h_2 is rechecked with the condition rule $y = lb_2$. If $y = lb_2$ is fulfilled, the y value is replaced with the y resulting from the label prediction with input h_3 obtained from the third CNN model (M_3). The final condition y obtained from h_1, h_2, and h_3 is checked with the rule $y = lb_3$. If the value meets these conditions, the value is replaced with the y value resulting from label prediction with input h_4 obtained from the fourth CNN model (M_4) (see Algorithm 2, lines 3 to 11). The algorithm's output is y as the final classification result by implementing the CNN model structure resulting from the mmC algorithm.

3.8 Performance evaluation model

To evaluate the performance of each single CNN model and multi-model of CNN constructed using the proposed algorithm, we used measures of accuracy (Acr), precision (Prs), sensitivity (Svt), specificity (Sft), and F-score ($F\text{-}sc$) which is obtained based on the true positive (Tp), false negative (Fn), true negative (Tn), and false positive (Fp) [37]. In this study, there are four labels in the dataset for testing, namely glioma, meningioma, pituitary, and no tumor. Accordingly, each label can use these performance indicators with the same definition. For example, for glioma, Tp is the number of times the MRI image of the glioma is labeled glioma based on classification results, Fn is the number of times the MRI image of the glioma brain is labeled other than glioma based on classification results, Tn is the number of times MRI image of other than glioma is labeled other than glioma, and Fp is the number of times MRI images of brains other than glioma is labeled glioma in the same way. For calculations Acr, Prs, Svt, Sft, and $F\text{-}sc$ are generally defined in Eqs. (5)-(9) which can be determined for each label [38, 39].

$$Acr = (Tp + Tn)/(Tp+Fp+Tn+Fn) \quad (5)$$

$$Prs = Tp/(Fp+Tp) \quad (6)$$

$$Svt = Tp/(Fn+Tp) \quad (7)$$

$$Sft = Tn/(Fp+Tn) \quad (8)$$

$$F\text{-}sc = \frac{2(Prs)(Svt)}{(Prs+Svt)} \quad (9)$$

4. Experiments

Before constructing the CNN multi-model, the training was carried out on each CNN model, namely Xception, DensNet-201, and EfficientNet-B3. This process involved datasets for training and validation, as shown in Table 2. In this study, we implemented all training and testing by using Google Colab.

Several CNN model parameters in this study are used during training apart from those previously explained, including input shape, number of epochs, batch size, optimizer, learning rate, and pre-training weights. The input shape in training or testing is 224x224x3, commonly used in these models. The number of epochs in testing includes 10, 20, and 30, thus, for the same model, there are three training outcomes. The batch size and learning rate measures for each training are 16 and 0.001 [12, 13, 40]. The optimizer used for each training is Adamax because it is more stable [41, 42]. In this study, because the training dataset is relatively small, transfer learning is carried out so that the initialization of the weights when training each model uses the weights resulting from training the model using the ImageNet dataset.

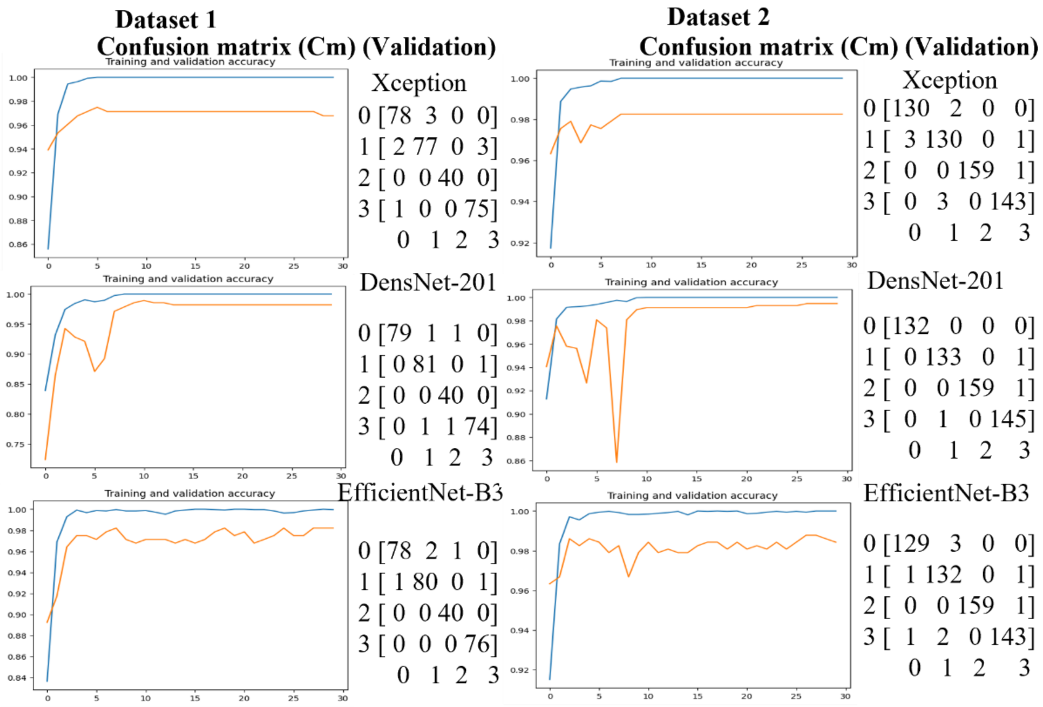
The construction of a multi-model CNN with the proposed algorithm (mmC) involving several CNN models (Xception, DensNet-201, and EfficientNet-B3) includes two scenarios: first, constructing a multi-model of the CNN from the same model but different epochs, and second, constructing a multi-model of the CNN from several different CNN models.

5. Results and discussion

5.1 Experimental results

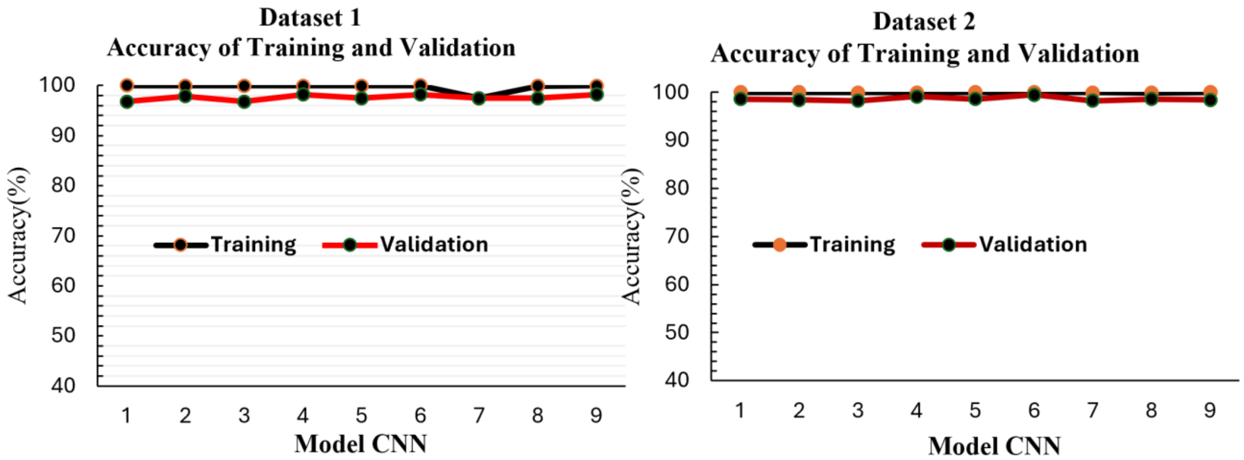
In this section, the experimental results reported are training and validation performance of a single CNN model, performance of a single CNN model on the testing dataset, and performance of CNN multi-model on the testing dataset.

Fig. 4 shows the training results of each CNN model (a single model) on Datasets 1 and 2. Each model obtained the training performance by involving transfer learning for each CNN model. In general, each model showed no tendency for overfitting to occur.



0 = glioma, 1 = meningioma, 2 = no tumor, 3 = pituitary

(a)



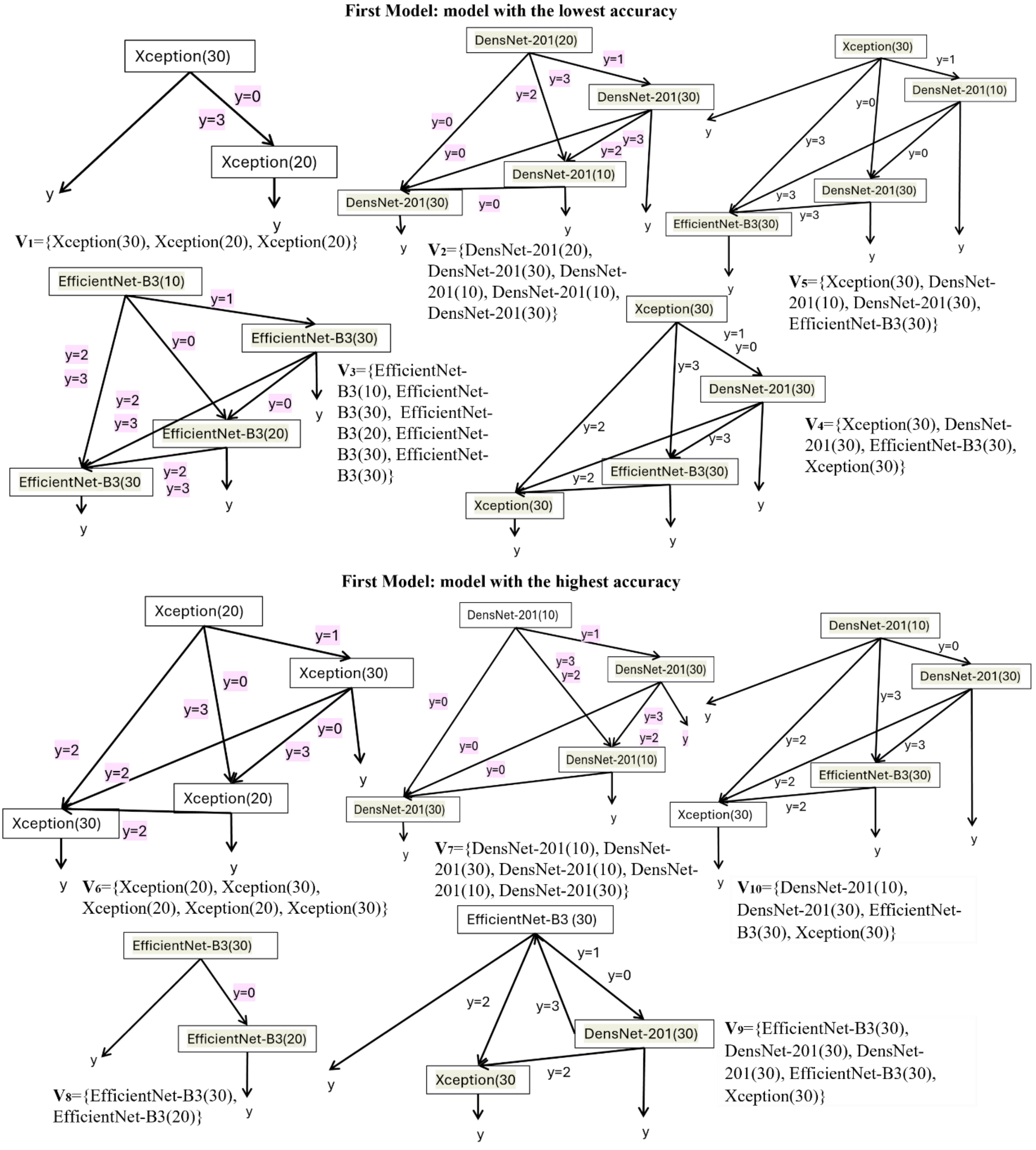
1=Xception(10), 2=Xception(20), 3= Xception(30), 4= DensNet-201(10), 5= DensNet-201(20), 6= DensNet-201(30),7= EfficientNet-B3(10), 8= EfficientNet-B3(20), 9= EfficientNet-B3(30)

(b)

Figure. 4 Training and validation: (a) confusion matrix of validation and (b) accuracy in the last epoch of each CNN model

Fig. 4 (b) shows that the deviation in training and validation accuracy in the last epoch is relatively tiny. These models obtained the training accuracy of Dataset 1 between 97.49%-100% and validation accuracy between 96.77%-98.21%. On Dataset 2, these models yielded a training accuracy of 99.92%-100%, while validation accuracy was 98.25%-99.48%. The validation confusion matrix in the last epoch of each model has an essential role in seeing the advantages between the trained CNN models.

Tables 3 and 4 are the test performance results of each CNN model on Datasets 1 and 2. Testing on Dataset 1 with 300 brain MRI images, the DensNet-201 model at epoch 10 provided the best accuracy compared to other single CNN models with a brain tumor classification accuracy of 96.45%. The next best accuracy was EfficientNet-B3 at epoch 30, which produced an accuracy of 96.13% and was better than Xception and DensNet-201 at epoch 30. EfficientNet-B3 obtained the lowest accuracy in this



V1={Xception(30), Xception(20), Xception(20)}

V2={DensNet-201(20), DensNet-201(30), DensNet-201(10), DensNet-201(10), DensNet-201(30)}

V5={Xception(30), DensNet-201(10), DensNet-201(30), EfficientNet-B3(30)}

V3={EfficientNet-B3(10), EfficientNet-B3(30), EfficientNet-B3(20), EfficientNet-B3(30), EfficientNet-B3(30)}

V4={Xception(30), DensNet-201(30), EfficientNet-B3(30), Xception(30)}

V6={Xception(20), Xception(30), Xception(20), Xception(20), Xception(30)}

V7={DensNet-201(10), DensNet-201(30), DensNet-201(10), DensNet-201(10), DensNet-201(30)}

V10={DensNet-201(10), DensNet-201(30), EfficientNet-B3(30), Xception(30)}

V9={EfficientNet-B3(30), DensNet-201(30), DensNet-201(30), EfficientNet-B3(30), Xception(30)}

V8={EfficientNet-B3(30), EfficientNet-B3(20)}

Figure. 5 CNN multi-model structures for brain tumor classification on testing Dataset 1

test at epoch 10, 93.55%. Based on the precision, sensitivity, specificity, and F-score values, DensNet-201 at epoch 10 was better than the other models. While EfficientNet-B3 at epoch 10 yielded the lowest performance compared to other single CNN models. For the test on Dataset 2, Xception at epoch 20 and EfficientNet-B3 at epoch 10 produced the best accuracy, precision, sensitivity, specificity, and F-score compared to other CNN models. Meanwhile, EfficientNet-B3 at epoch 20 produced the lowest performance.

The performance of the test results for all single CNN models on the test dataset (Dataset 1 and Dataset 2) produced relatively different dispersion. In testing Dataset 1, the difference in tumor classification accuracy for the model with the highest and lowest accuracy was 2.9% (96.45%- 93.55%), greater than testing on Dataset 2, 0.39% (99.47% - 99.08%). The training sample size on Datasets 1 and 2 influenced the resulting classification performance.

Received: June 20, 2024. Revised: July 28, 2024.

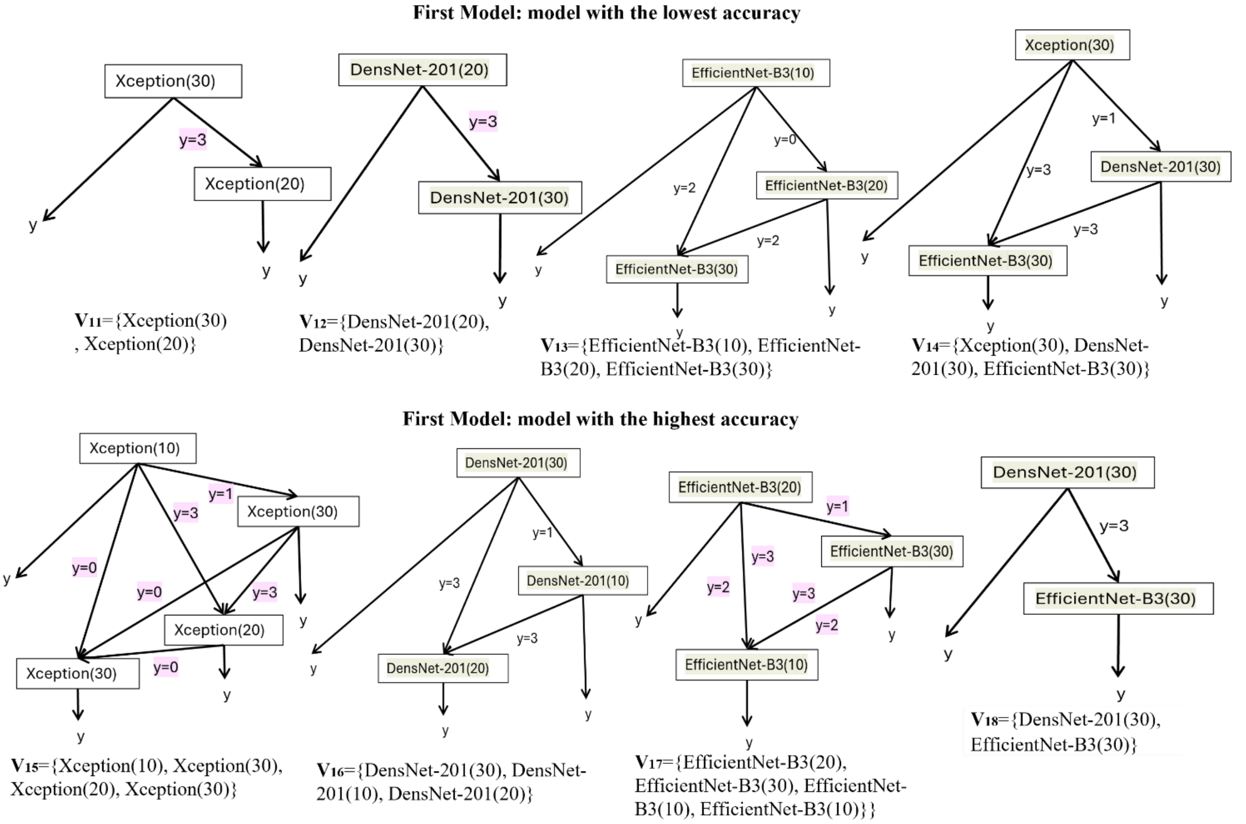
On the other hand, the number of training epochs in each CNN model test was not directly proportional to the resulting classification performance.

In this study, the construction of a multi-model CNN with the proposed algorithm (mmC) involving several CNN models (Xception, DensNet-201, and EfficientNet-B3) includes two scenarios: first, constructing a multi-model of the CNN from the same model but different epochs, and second, constructing a multi-model of the CNN from several different CNN models. Tables 3 and 4 are the summaries of the performance results of CNN multi-model in classifying brain tumors on testing Datasets 1 and 2.

The test results on Dataset 1 for the first scenario with the first model in the structure are the models that have the lowest validation accuracy, shown in Table 3 and Fig. 5. The multi-model of CNN constructed from the Xception model at epochs 20 and 30 ($G_1(V_1, E_1)$) produced 96.45% accuracy. Although the multi-model performance is not the best among other multi-models, it is still better than the single Xception model. The multi-model of G_2 was constructed by three training models, namely the DensNet-201 model at epochs 10, 20, and 30, producing an accuracy of 97.10%. With these results, G_2 performed better than the single DensNet-201 model at each epoch and better than the multi-model of G_1. The lowest classification accuracy, namely 96.13%, was produced by the multi-model of G_3, which the EfficientNet-B3 model constructed at epochs 10, 20, and 30. This result was similar in accuracy to EfficientNet-B3 at epoch 30. This condition occured because the difference in the accuracy of the models constructing the G_3 structure was relatively significant, or two models constructing the G_3 had low accuracy compared to the model with maximum accuracy.

In the second scenario, CNN's multi-models, constructed from different single models, provided good results, such as G_4 and G_5. G_5 provided the best performance compared to the others, having the greatest accuracy (97.74%) and the best precision, sensitivity, specificity, and F-score values. The CNN models constructing G_5 are Xception, DensNet-201, EfficientNet-B3 at epoch 30, and DensNet-201 at epoch 10.

The multi-model structure is shown by $G_5(V_5, E_5)$ with V_5 = {(3), (4), (6), (9)} = {Xception (30), DensNet-201 (10), DensNet-201(30), EfficientNet-B3 (30)}, and E_5 ={1, 0, 3} which can be visually seen in Fig. 5 for the corresponding V. All of these models for constructing G_5 had relatively the best accuracy among other single models. Therefore, there is more potential to combine the advantages of each model based on accuracy and Fp values.



First Model: model with the lowest accuracy

$\mathbf{V_{11}}$={Xception(30) , Xception(20)}

$\mathbf{V_{12}}$={DensNet-201(20), DensNet-201(30)}

$\mathbf{V_{13}}$={EfficientNet-B3(10), EfficientNet-B3(20), EfficientNet-B3(30)}

$\mathbf{V_{14}}$={Xception(30), DensNet-201(30), EfficientNet-B3(30)}

First Model: model with the highest accuracy

$\mathbf{V_{15}}$={Xception(10), Xception(30), Xception(20), Xception(30)}

$\mathbf{V_{16}}$={DensNet-201(30), DensNet-201(10), DensNet-201(20)}

$\mathbf{V_{17}}$={EfficientNet-B3(20), EfficientNet-B3(30), EfficientNet-B3(10), EfficientNet-B3(10)}}

$\mathbf{V_{18}}$={DensNet-201(30), EfficientNet-B3(30)}

Figure. 6 CNN multi-model structures for brain tumor classification on testing Dataset 2

Table 3. Performance of brain tumor classification by a single CNN model and multi-model of CNN on testing Dataset 1

Model (Epoch)		***Acr***	**Average (%)**				**Δ**
		(%)	***Prs***	***Svt***	***Sft***	***F-sc***	(%)*
Single Model	(1) Xception(10)	94.52	95.53	95.28	98.32	95.36	3.23
	(2) Xception(20)	94.84	94.29	95.59	98.29	94.83	2.90
	(3) Xception(30)	95.81	96.08	96.39	98.54	96.22	1.94
	(4) DensNet-201(10)	96.45	96.63	96.96	98.78	96.76	1.29
	(5) DensNet-201(20)	95.16	94.98	95.55	98.37	95.21	2.58
	(6) DensNet-201(30)	95.81	96.08	96.37	98.55	96.21	1.94
	(7) EfficientNet-B3(10)	93.55	93.09	94.46	97.85	93.63	4.19
	(8) EfficientNet-B3(20)	94.84	93.96	95.59	98.33	94.58	2.90
	(9) EfficientNet-B3(30)	96.13	95.64	96.69	98.72	96.11	1.61
Multi-Model of CNN**	$G_1(V_1,E_1),V_1=\{(3),(2),(2)\}, E_1=\{0,3\}$	96.45	96.67	96.96	98.77	96.78	1.29
	$G_2(V_2,E_2),V_2=\{(5),(6),(4),(4),(6)\}, E_2=\{1,3,2,0\}$	**97.10**	**97.20**	**97.51**	**99.00**	**97.33**	0.65
	$G_3(V_3,E_3),V_3=\{(7),(9),(8),(9),(9)\}, E_3=\{1,0,2,3\}$	96.13	95.64	96.68	98.72	96.11	1.61
	$G_4(V_4,E_4),V_4=\{(3),(6),(6),(9),(3)\}, E_4=\{1,0,3,2\}$	**97.10**	**97.20**	**97.51**	**99.00**	**97.35**	0.65
	$G_5(V_5,E_5),V_5=\{(3),(4),(6),(9)\}, E_5=\{1,0,3\}$	**97.74**	**97.78**	**98.06**	**99.22**	**97.92**	0.00
Multi-Model of CNN***	$G_6(V_6,E_6),V_6=\{(2),(3),(2),(2),(3)\}, E_6=\{1,0,3,2\}$	96.13	96.35	96.68	98.66	96.49	1.61
	$G_7(V_7,E_7),V_7=\{(4),(6),(4),(4),(6)\}, E_7=\{1,3,2,0\}$	**97.10**	**97.20**	**97.51**	**99.01**	**97.33**	0.65
	$G_8(V_8,E_8),V_8=\{(9),(8)\}, E_8=\{0\}$	96.13	95.64	96.68	98.72	96.10	1.61
	$G_9(V_9,E_9),V_9=\{(9),(6),(6),(9),(3)\}, E_9=\{1,0,3,2\}$	**97.10**	**97.20**	**97.51**	**99.00**	**97.35**	0.65
	$G_{10}(V_{10},E_{10}), V_{10}=\{(4),(6),(9),(3)\}, E_{10}=\{0,3,2\}$	**97.42**	**97.51**	**97.79**	**99.11**	**97.64**	0.32

*) Δ=max. accuracy of multi-model - all model, **) Proposed Algorithm, First Model = model with the lowest accuracy of validation, ***) Proposed Algorithm, First Model=model with highest accuracy of validation

Table 4. Performance of brain tumor classification by a single CNN model and multi-model of CNN on testing Dataset 2.

Model (Epoch)		***Acr***	**Average (%)**				**Δ**
		(%)	***Prs***	***Svt***	***Sft***	***F-sc***	(%)*
Single Model	(1) Xception(10)	99.24	99.20	99.17	99.75	99.19	0.46
	(2) Xception(20)	99.47	99.43	99.42	99.83	99.42	0.23
	(3) Xception(30)	99.39	99.35	99.38	99.80	99.36	0.31
	(4) DensNet-201(10)	99.16	99.10	99.13	99.73	99.12	0.53
	(5) DensNet-201(20)	99.39	99.36	99.34	99.80	99.35	0.31
	(6) DensNet-201(30)	99.39	99.34	99.34	99.80	99.34	0.31
	(7) EfficientNet-B3(10)	99.47	99.42	99.42	99.83	99.42	0.23
	(8) EfficientNet-B3(20)	99.08	99.06	99.05	99.70	99.05	0.61
	(9) EfficientNet-B3(30)	99.31	99.30	99.26	99.77	99.28	0.38
Multi-Model of CNN**	$G_{11}(V_{11},E_{11}),V_{11}=\{(3),(2)\}, E_{11}=\{3\}$	**99.54**	**99.51**	**99.54**	**99.85**	**99.53**	0.15
	$G_{12}(V_{12},E_{12}),V_{12}=\{(5),(6)\}, E_{12}=\{3\}$	**99.54**	**99.53**	**99.50**	**99.85**	**99.52**	0.15
	$G_{13}(V_{13},E_{13}),V_{13}=\{(7),(8),(9)\}, E_{13}=\{0,2\}$	99.47	99.43	99.42	99.83	99.42	0.23
	$G_{14}(V_{14},E_{14}),V_{14}=\{(3),(6),(9)\}, E_{14}=\{1,3\}$	**99.69**	**99.67**	**99.67**	**99.90**	**99.67**	0.00
Multi-Model of CNN***	$G_{15}(V_{15},E_{15}),V_{15}=\{(1),(3),(2),(3)\}, E_{15}=\{1,3,0\}$	**99.54**	**99.53**	**99.50**	**99.85**	**99.51**	0.15
	$G_{16}(V_{16},E_{16}),V_{16}=\{(6),(4),(5)\}, E_{16}=\{1,3\}$	**99.54**	**99.50**	**99.51**	**99.85**	**99.50**	0.15
	$G_{17}(V_{17},E_{17}),V_{17}=\{(8),(9),(7),(7)\}, E_{17}=\{1,3,2\}$	**99.62**	**99.59**	**99.59**	**99.88**	**99.59**	0.08
	$G_{18}(V_{18},E_{18}),V_{18}=\{(6),(9)\}, E_{18}=\{3\}$	**99.62**	**99.59**	**99.59**	**99.88**	**99.59**	0.08

*) Δ=max. accuracy of multi-model - all model, **) Proposed Algorithm, First Model = model with the lowest accuracy of validation, ***) Proposed Algorithm, First Model=model with highest accuracy of validation

For the first scenario where the first model in the structure is the model that has the highest accuracy, the multi-model of G_6 constructed by the Xception model with epochs 10, 20, and 30 produced better accuracy performance than a single model Xception and other single models, although not all. In contrast, the multi-model of G_7 constructed from the DensNet-201 model at epochs 10, 20, and 30 obtained better performance than all single models at all epochs. On the other hand, the multi-model of G_8 had the same performance as the multi-model of G_3 and had the same problems. Meanwhile, the mmC algorithm with different model inputs yielded the multi-models of G_9 and G_{10} in the second scenario. Both multi-models were able to improve tumor classification performance not only in accuracy but also in precision, sensitivity, specificity, and F-score. G_{10} was the best in this scenario, with an accuracy of

Received: June 20, 2024. Revised: July 28, 2024.

97.42%. G_{10} (V_{10}, E_{10}) is a multi-model with V_{10} = {(4), (6), (9), (3)} = {DensNet-201(10), DensNet-201(30), EfficientNet-B3(30), Xception(30)} and E_{10} ={0, 3, 2}, which are visually shown in Fig. 5 for the corresponding *V*.

Furthermore, a summary of the performance of CNNs multi-model in brain tumor classification for all scenarios in testing Dataset 2 is shown in Table 4. In contrast, the structure of each multi-model can be seen in Fig. 6. In the first scenario, with the first model having the lowest validation accuracy, a multi-model of CNN constructed from the Xception model at epochs 20 and 30 (G_{11}(V_{11}, E_{11})) produced an accuracy of 99.54%. With this performance, even though G_{11} is not the best among other multi-models, it is still better than all single models at every epoch. On the other hand, G_{12}, which the DensNet-201 model constructed at epochs 20 and 30, also obtained the same accuracy performance. Meanwhile, the lowest accuracy among the multi-model of CNNs was G_{13}, which the EfficientNet-B3 model constructed at epochs 10, 20, and 30. The difference between the lowest accuracy of the multi-model structure-constructing model and the model with maximum accuracy is relatively significant. Hence, the advantages of the model with maximum accuracy have yet to cover the shortcomings of the model with the lowest accuracy.

In the second scenario, the mmC algorithm obtained a multi-model of CNN G_{14}(V_{14}, E_{14}) with V_{14} = {(3), (6), (9)} = {Xception (30), DensNet-201 (30), EfficientNet-B3 (30)} and E_{14} ={1,3}. G_{14} produced the best accuracy among all multi-model CNNs, namely 99.69%, and also obtained the best precision, sensitivity, specificity, and F-score among other multi-models.

For the first scenario with the first model with the highest validation accuracy, the mmC algorithm yielded multi-models of G_{15}, G_{16}, and G_{17}. All CNN multi-models performed better than single models in all epochs, as shown in Table 4. Meanwhile, in the second scenario, it produced G_{18} with V_{18} = {DensNet-201(30), EfficientNet-B3(30)} and E_{18} ={3}, which were able to increase the accuracy of tumor classification for all single CNN models. Apart from that, the multi-model could also improve precision, sensitivity, specificity, and F-score.

5.2 Discussion

In this section, we investigate the performance of the multi-model of CNN by using mmC Algorithm.

From the test results on Datasets 1 and 2, almost all CNN multi-models could provide improved accuracy to single CNN models. Testing on testing Dataset 1, nearly all CNN multi-models constructed with the mmC algorithm resulted in improved accuracy to the single model that formed them. The multi-model of G_1 provided improved classification accuracy over all single CNN models at all epochs apart from DensNet-201 at epoch 10 of 0.32%-2.9%. Multi-models of G_2, G_4, G_7, and G_9 improved classification accuracy over all single CNN models by 0.65%-3.55%. G_{10} provided accuracy improvements for all single models of 0.97%-3.87%. The multi-model of G_5 was the best among other CNN multi-models and provided improved accuracy in brain tumor classification overall single models of 1.29%-4.19%. Almost all CNN multi-models on testing Dataset 2 also demonstrated improvements in brain tumor classification performance. Multi-models of G_{11}, G_{12}, G_{15}, and G_{16} improved classification accuracy by 0.07%-0.46% for all single CNN models. The multi-models of G_{17} and G_{18} improved classification accuracy over all single CNN models, 0.15%-0.54%. Meanwhile, G_{14} was the best multi-model of CNN, providing improvements over all single CNN models of 0.22%-0.61%.

Judging from the best performance obtained by the single CNN model and multi-model of CNN, on testing Dataset 1, DensNet-201 at epoch 10 produced the best performance among other single models with an accuracy of 96.45%. Meanwhile, the multi-model of G_5 produced the best performance, with an accuracy of 97.74%. Table 5 shows in more detail the comparison between the two. G_5 improved the precision of meningioma and pituitary classification, glioma and meningioma classification sensitivity, and the specificity of meningioma and pituitary classification. When viewed from the F-score, G_5 provided improved classification F-score values for glioma, meningioma, and pituitary. For testing Dataset 2, the single model that gave the best performance among the other single models was Xception at epoch 20 and EfficientNet-B3 at epoch 10. Meanwhile, G_{14}'s multi-model yielded the best performance against all of them. G_{14} provided improved meningioma precision, glioma sensitivity, and meningioma specificity. Meanwhile, looking at the F-score, G_{14} gave an increased F-score for glioma and meningioma (see Table 5).

Several previous studies also evaluated the proposed method using the testing Dataset 2, a combination of Figshare, SARTAJ, and Br35H datasets. Shilaskar et al. [22] proposed the HOG-XG Boost method and obtained a classification accuracy of 92.02%. Atha & Chaki [23] proposed the SSBTCNet model and yielded a tumor classification accuracy of 96.5%. At the same time, Rasheed et al. [1] proposed image enhancement and CNN.

Table 5. Best performances of a single CNN model and multi-model of CNN.

Model	Class/Label	Confusion Matrix (*Cm*)	*Prs* (%)	*Svt* (%)	*Sft* (%)	*F-sc* (%)
Testing Dataset 1						
DensNet-201 (10)	Glioma	[84 4 1 1]	**97.67**	93.33	**99.09**	95.45
	Meningioma	[2 86 0 3]	95.56	94.51	98.17	95.03
	No Tumor	[0 0 44 0]	97.78	100	99.62	98.88
	Pituitary	[0 0 0 85]	95.51	100	98.22	97.70
Multi-Model of CNN $G_5(V_5,E_5),V_5$={ Xception(30), DensNet-201(10), DensNet-201(30), EfficientNet-B3(30)}, E_5={1,0,3}	Glioma	[86 3 1 0]	96.63	**95.56**	98.64	**96.09**
	Meningioma	[3 88 0 0]	**96.70**	**96.70**	**98.63**	**96.70**
	No Tumor	[0 0 44 0]	97.78	100	99.62	98.88
	Pituitary	[0 0 0 85]	**100**	100	**100**	**100**
Testing Dataset 2						
Xception(20)	Glioma	[296 4 0 0]	99.33	98.67	**99.80**	99.00
	Meningioma	[2 304 0 0]	98.38	**99.35**	99.50	98.86
	No Tumor	[0 0 405 0]	100	100	100	100
	Pituitary	[0 1 0 299]	100	99.67	100	99.83
EfficientNet-B3(10)	Glioma	[297 3 0 0]	99.00	99.00	99.70	99.00
	Meningioma	[3 303 0 0]	98.70	99.02	99.60	98.86
	No Tumor	[0 0 405 0]	100	100	100	100
	Pituitary	[0 1 0 299]	100	99.67	100	99.83
Multi-Model of CNN $G_{14}(V_{14},E_{14}),V_{14}$={Xception(30), DensNet-201(30),EfficientNet-B3(30)}, E_{14}={1,3}	Glioma	[300 0 0 0]	99.01	**100**	99.70	**99.50**
	Meningioma	[3 303 0 0]	**99.67**	99.02	**99.90**	**99.34**
	No Tumor	[0 0 405 0]	100	100	100	100
	Pituitary	[0 1 0 299]	100	99.67	100	99.83

Table 6. Comparison of multi-model of CNN with the existing methods

No.	Authors	Methods	*Acr* (%)	
			Testing Dataset 1	**Testing Dataset 2**
1	This Work, Original Model in [15]	Xception (transfer learning, customization of top layer)	95.81	99.39
2	This Work, Original Model in [16]	DensNet-201(transfer learning, customization of top layer)	96.45	99.39
3	This Work, Original Model in [17]	EfficientNet-B3(transfer learning, customization of top layer)	96.13	99.47
4	Shilaskar et al. [22]	HOG-XG Boost	-	92.02
5	Atha & Chaki [23]	SSBTCNet model	-	96.5
6	Rasheed et al. [1]	Image enhancement and CNN	-	97.84
7	Proposed Methods	mmC (Xception, DensNet-201, EfficientNet-B3}	**97.74**	**99.69**

Their method achieved a classification accuracy of 97.84%. All these results show that our proposed method produces more classification accuracy than all (see Table 6).

The HOG-XG Boost method in [22] implemented a histogram of oriented gradients (HOG) to extract the feature and conducted it separately from the classifier (XG Boost). With its oriented gradient histogram, HOG will not get the representative features for tumors that have high variations of shapes, sizes, and tumor positions in MRI images. Glioma and meningioma tumors, in some cases, are highly similar that HOG will produce features that are difficult for the classifier (XG Boost) to differentiate. In addition, HOG is used separately from the classifier (XG Boost). Thus, there is no guarantee that the feature extraction results are suitable or representative for the classifier to be used. Meanwhile, SSBTCNet in [23] combines an unsupervised autoencoder with a supervised classification network and uses it to avoid extracting completely separated features from the classifier. Even though it can help learn hidden descriptors, there are types of brain tumors that are very similar in size and shape (for example, glioma and meningioma), so unlabeled data in training with SSBTCNet can produce errors for the supervised classification network. Additionally, since it does not combine several models, this method will obtain less than optimal performance in classifying types of brain tumors with very varied sizes and shapes. Likewise, using only a CNN model in [1] to distinguish tumor types with varying sizes, shapes, and positions would also be difficult despite image enhancement in the previous stage. Meanwhile, the multi-model CNN in this study, which was constructed using the proposed algorithm (mmC), combined the advantages of each CNN model used (Xceptian, DensNet-201, and EfficientNet-B3). The benefits of this model are determined based on the lowest false positive validation value among brain tumor types and cover the shortcomings of other CNN models. mmC constructs a multi-model CNN structure by combining the advantages of each CNN model and leaving behind the disadvantages to improve brain tumor classification performance.

6. Conclusion

A multi-model of CNN for the classification of brain tumors based on brain MRI images is one solution to improve classification performance by combining the advantages of each single CNN model that constructs it. The mmC algorithm proposed in this study works by combining the benefits of several CNN models based on validation accuracy values and false positives of validation. Test results using Dataset 1, the mmC algorithm produced the best multi-model of CNN, namely $G_5(V_5, E_5)$ with V_5 ={Xception(30), DensNet-201(10), DensNet-201 (30), EfficientNet-B3 (30)}, and E_5 ={1, 0, 3}. G_5 yielded classification accuracy, precision, sensitivity, specificity, and F-score of 97.74%, 97.78%, 98.06%, 99.22%, and 97.92%, respectively. Meanwhile, with Dataset 2, the algorithm produced the best multi-model CNN, namely $G_{14}(V_{14}, E_{14})$ with V_{14} = {Xception (30), DensNet-201 (30), EfficientNet-B3(30)}, and E_{14} ={1, 3}. G_{14} obtained accuracy, precision, sensitivity, specificity, and F-score values in brain tumor classification of 99.69%, 99.67%, 99.67%, 99.90%, and 99.67%, respectively.

With these results, the proposed algorithm has produced the new CNN multi-model, which has excellent potential to help medical personnel classify types of brain tumors with high accuracy, therefore, that they can take appropriate follow-up action. For clinical implementation, increasing the amount of training and testing data to improve classification performance is necessary.

Conflicts of Interest

The authors declare no conflict of interest.

Author Contributions

Conceptualization, Irwan Budi Santoso; methodology, Irwan Budi Santoso, software, Irwan Budi Santoso; validation, Irwan Budi Santoso, formal analysis, Irwan Budi Santoso, Supriyono, and Shoffin Nahwa Utama; investigation, Irwan Budi Santoso and Supriyono; resources, Irwan Budi Santoso; data curation, Irwan Budi Santoso and Supriyono; writing—original draft preparation, Irwan Budi Santoso; writing—review and editing, Irwan Budi Santoso, Supriyono, and Shoffin Nahwa Utama; supervision, Irwan Budi Santoso; project administration, Irwan Budi Santoso; funding acquisition, Irwan Budi Santoso.

Acknowledgments

This research was financially supported by Universitas Islam Negeri Maulana Malik Ibrahim Malang through the research scheme of the National Development Applied Research (Penelitian Terapan Pengembangan Nasional) with contract number 1143A/LP2M/TL.00/02/2024.

References

[1] Z. Rasheed, Y. K. Ma, I.Ullah, Y. Y.Ghadi, M. Z.Khan, M. A.Khan, A.Abdusalomov, F.Alqahtani, and A. M. Shehata, “Brain Tumor Classification from MRI Using Image Enhancement and Convolutional Neural Network Techniques”, *Brain Sciences*, Vol. 13, No. 9, 2023.

[2] E. A. S. El-Dahshan, H. M. Mohsen, K. Revett, and A. B. M. Salem, “Computer-aided diagnosis of human brain tumor through MRI: A survey and a new algorithm”, *Expert Systems with Applications*, Vol. 41, No. 11, pp. 5526-5545, 2014.

[3] N. Varuna Shree and T. N. R. Kumar, “Identification and classification of brain tumor MRI images with feature extraction using DWT and probabilistic neural network”, *Brain Informatics*, Vol. 5, No. 1, pp. 23-30, 2018.

[4] J. Kang, Z. Ullah, and J. Gwak, “Mri-based brain tumor classification using ensemble of deep features and machine learning classifiers”, *Sensors*, Vol. 21, No. 6, pp. 1-21, 2021.

[5] S. Ahmad and P. K. Choudhury, “On the Performance of Deep Transfer Learning Networks for Brain Tumor Detection Using MR Images”, *IEEE Access*, Vol. 10, pp. 59099-59114, 2022.

[6] S. Shanthi, S. Saradha, J. A. Smitha, N. Prasath, and H. Anandakumar, “An efficient automatic

brain tumor classification using optimized hybrid deep neural network", *International Journal of Intelligent Networks*, Vol. 3, pp. 188-196, 2022.

[7] M. Rizwan, A. Shabbir, A. R. Javed, M. Shabbir, T. Baker, and D. Al-Jumeily Obe, "Brain Tumor and Glioma Grade Classification Using Gaussian Convolutional Neural Network", *IEEE Access*, Vol. 10, pp. 29731-29740, 2022.

[8] A. S. Musallam, A. S. Sherif, and M. K. Hussein, "A New Convolutional Neural Network Architecture for Automatic Detection of Brain Tumors in Magnetic Resonance Imaging Images", *IEEE Access*, Vol. 10, pp. 2775-2782, 2022.

[9] S. Chatterjee, F. A. Nizamani, A. Nürnberger, and O. Speck, "Classification of brain tumours in MR images using deep spatiospatial models", *Scientific Reports*, Vol. 12, No. 1, pp. 1-11, 2022.

[10] M. S. I. Khan, A. Rahman, T. Debnath, M. R. Karim, M. K.Nasir, S. S. Band, A. Mosavi, and I. Dehzangi, "Accurate brain tumor detection using deep convolutional neural network", *Computational and Structural Biotechnology Journal*, Vol. 20, pp. 4733-4745, 2022.

[11] N. Noreen, S. Palaniappan, A. Qayyum, I. Ahmad, M. Imran, and M. Shoaib, "A Deep Learning Model Based on Concatenation Approach for the Diagnosis of Brain Tumor", *IEEE Access*, Vol. 8, pp. 55135-55144, 2020.

[12] I. B. Santoso, S. N. Utama, and Supriyono, "A New Voting of Convolutional Neural Networks for Brain Tumor Detection Based on MRI Images", *International Journal of Intelligent Engineering and Systems*, Vol. 17, No. 1, pp. 212-227, 2024, doi: 10.22266/ijies2024.0229.21.

[13] I. B. Santoso, Y. Adrianto, A. D. Sensusiati, D. P. Wulandari, and I. K. E. Purnama, "Ensemble Convolutional Neural Networks With Support Vector Machine for Epilepsy Classification Based on Multi-Sequence of Magnetic Resonance Images", *IEEE Access*, Vol. 10, pp. 32034-32048, 2022.

[14] T. A. Jemimma and Y. Jacob Vetharaj, "A Survey on Brain Tumor Segmentation and Classification", *International Journal of Software Innovation*, Vol. 10, No. 1, 2022.

[15] F. Chollet, "Xception: Deep learning with depthwise separable convolutions", In: *Proc. of 30th IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, USA, pp. 1800-1807, 2017.

[16] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks", In: *Proc. of 30th IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, USA, pp. 2261-2269, 2017.

[17] M. Tan and Q. V. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks", In: *Proc. of 36th International Conference on Machine Learning (ICML)*, Long Beach, California, USA, pp. 10691-10700, 2019.

[18] S. Asif, W. Yi, Q. U. Ain, J. Hou, T. Yi, and J. Si, "Improving Effectiveness of Different Deep Transfer Learning-Based Models for Detecting Brain Tumors From MR Images", *IEEE Access*, Vol. 10, pp. 34716-34730, 2022.

[19] D. Reyes and J. Sánchez, "Performance of convolutional neural networks for the classification of brain tumors using magnetic resonance imaging", *Heliyon*, Vol. 10, No. 3, pp. e25468, 2024.

[20] A. Younis, L. Qiang, C. O. Nyatega, M. J. Adamu, and H. B. Kawuwa, "Brain Tumor Analysis Using Deep Learning and VGG-16 Ensembling Learning Approaches", *Applied Sciences*, Vol. 12, No. 14, 2022.

[21] W. Jun and Z. Liyuan, "Brain Tumor Classification Based on Attention Guided Deep Learning Model", *International Journal of Computational Intelligence Systems*, Vol. 15, No. 1, pp. 1-9, 2022.

[22] S. Shilaskar, T. Mahajan, S. Bhatlawande, S. Chaudhari, R. Mahajan, and K. Junnare, "Machine Learning based Brain Tumor Detection and Classification using HOG Feature Descriptor", In: *Proc. of 2023 International Conference on Sustainable Computing and Smart Systems (ICSCSS)*, Coimbatore, pp. 67-75, 2023.

[23] Z. Atha and J. Chaki, "SSBTCNet: Semi-Supervised Brain Tumor Classification Network", *IEEE Access*, Vol. 11, pp. 141485-141499, 2023.

[24] Thomasdubail, *Brain tumors 256x256*, 2023. Available online at: https://www.kaggle.com /datasets/thomasdubail/brain-tumors-256x256

[25] S. Bhuvaji, A. Kadam, P. Bhumkar, S. Dedge, and S. Kanchan, *Brain Tumor Classification (MRI)*, 2020. Available online at: https://www.kaggle.com/datasets/sartajbhuvaji/ brain-tumor-classification-mri

[26] M. Nickparvar, *Brain Tumor MRI Dataset*, 2021. Available online at: https://www.kaggle.com/datasets/masoudnickp arvar/brain-tumor-mri-dataset

[27] J. Cheng, *Brain Tumor Dataset Figshare*, 2017. Available online at: https://figshare.com/articles /dataset/brain_tumor_dataset/1512427

[28] A. Hamada, *Br35H Brain Tumor Detection 2020*, 2020. Available online at: https://www.kaggle.com/datasets/ahmedhamada0/brain-tumor-detection

[29] J. Deng, W. Dong, R. Socher, L.-J. Li, Kai Li, and Li Fei-Fei, “ImageNet: A large-scale hierarchical image database”, In: *Proc. of 2009 IEEE Conference on Computer Vision and Pattern Recognition*, Miami, FL, USA, pp. 248-255, 2009.

[30] A. Patil and M. Rane, “Convolutional Neural Networks: An Overview and Its Applications in Pattern Recognition”, *Smart Innovation, Systems and Technologies*, Vol. 195, pp. 21-30, 2021.

[31] M. Lin, Q. Chen, and S. Yan, “Network in network”, In: *Proc. of 2nd International Conference on Learning Representations (ICLR)*, Banff, AB, pp. 1-10, 2014.

[32] H. Alhichri, A. S. Alswayed, Y. Bazi, N. Ammour, and N. A. Alajlan, “Classification of Remote Sensing Images Using EfficientNet-B3 CNN Model with Attention”, *IEEE Access*, Vol. 9, pp. 14078-14094, 2021.

[33] P. P. Malla, S. Sahu, and A. I. Alutaibi, “Classification of Tumor in Brain MR Images Using Deep Convolutional Neural Network and Global Average Pooling”, *Processes*, Vol. 11, No. 3, pp. 1-17, 2023.

[34] Mp. Lakshmi, “Image Classification Using Machine Learning Techniques”, *International Journal of Advanced Research*, Vol. 7, No. 5, pp. 1238-1245, 2019.

[35] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, Cambridge, MA, USA: MIT Press, 2016.

[36] I. B. Santoso, Y. Adrianto, A. Sensusiati, D. Wulandari, and I. Purnama, “Epileptic EEG Signal Classification Using Convolutional Neural Network Based on Multi-Segment of EEG Signal”, *International Journal of Intelligent Engineering and Systems*, Vol. 14, No. 3, pp. 160-176, 2021, doi: 10.22266/ijies2021.0630.15.

[37] Y. Jiao and P. Du, “Performance measures in evaluating machine learning based bioinformatics predictors for classifications”, *Quantitative Biology*, Vol. 4, No. 4, pp. 320-330, 2016.

[38] H. Dalianis, *Clinical text mining: Secondary use of electronic patient records*, Springer Cham, pp.45-53, 2018.

[39] S. A. Hicks, I. Strümke, V. Thambawita, M. Hammou, M. A. Riegler, P. Halvorsen, and S. Parasa, “On evaluation metrics for medical applications of artificial intelligence”, *Scientific Reports*, Vol. 12, No. 1, pp. 1-9, 2022.

[40] S. Taheri, Z. Golrizkhatami, A. A. Basabrain, and M. S. Hazzazi, “A Comprehensive Study on Classification of Breast Cancer Histopathological Images: Binary Versus Multi-Category and Magnification-Specific Versus Magnification-Independent”, *IEEE Access*, Vol. 12, pp. 50431-50443, 2024.

[41] D. P. Kingma and J. L. Ba, “Adam: A method for stochastic optimization”, In: *Proc. of 3rd International Conference on Learning Representations (ICLR)*, pp. 1-15, 2015.

[42] B. Xiao, Y. Liu, and B. Xiao, “Accurate state-of-charge estimation approach for lithium-ion batteries by gated recurrent unit with ensemble optimizer”, *IEEE Access*, Vol. 7, pp. 54192-54202, 2019.