diagnostics

MDPI

Article

Investigating the Impact of Two Major Programming Environments on the Accuracy of Deep Learning-Based Glioma Detection from MRI Images

Vadi Su Yilmaz [1], Metehan Akdag [2], Yaser Dalveren [1], Resat Ozgur Doruk [1], Ali Kara [3,*] and Ahmet Soylu [4]

[1] Department of Electrical and Electronics Engineering, Atilim University, Kizilcasar Mahallesi, Incek Golbasi, Ankara 06830, Turkey
[2] Fonet Information Technologies, Kizilirmak Mahallesi, Cukurambar Cankaya, Ankara 06520, Turkey
[3] Department of Electrical and Electronics Engineering, Gazi University, Eti Mahallesi, Yukselis Sokak, Maltepe, Ankara 06570, Turkey
[4] Department of Computer Science, OsloMet—Oslo Metropolitan University, Pilestredet 35, Oslo 0167, Norway
* Correspondence: akara@gazi.edu.tr

Abstract: Brain tumors have been the subject of research for many years. Brain tumors are typically classified into two main groups: benign and malignant tumors. The most common tumor type among malignant brain tumors is known as glioma. In the diagnosis of glioma, different imaging technologies could be used. Among these techniques, MRI is the most preferred imaging technology due to its high-resolution image data. However, the detection of gliomas from a huge set of MRI data could be challenging for the practitioners. In order to solve this concern, many Deep Learning (DL) models based on Convolutional Neural Networks (CNNs) have been proposed to be used in detecting glioma. However, understanding which CNN architecture would work efficiently under various conditions including development environment or programming aspects as well as performance analysis has not been studied so far. In this research work, therefore, the purpose is to investigate the impact of two major programming environments (namely, MATLAB and Python) on the accuracy of CNN-based glioma detection from Magnetic Resonance Imaging (MRI) images. To this end, experiments on the Brain Tumor Segmentation (BraTS) dataset (2016 and 2017) consisting of multiparametric magnetic MRI images are performed by implementing two popular CNN architectures, the three-dimensional (3D) U-Net and the V-Net in the programming environments. From the results, it is concluded that the use of Python with Google Colaboratory (Colab) might be highly useful in the implementation of CNN-based models for glioma detection. Moreover, the 3D U-Net model is found to perform better, attaining a high accuracy on the dataset. The authors believe that the results achieved from this study would provide useful information to the research community in their appropriate implementation of DL approaches for brain tumor detection.

Keywords: brain tumor detection; glioma; deep learning; U-Net; V-Net; MATLAB; Python; performance assessment

Citation: Yilmaz, V.S.; Akdag, M.; Dalveren, Y.; Doruk, R.O.; Kara, A.; Soylu, A. Investigating the Impact of Two Major Programming Environments on the Accuracy of Deep Learning-Based Glioma Detection from MRI Images. *Diagnostics* **2023**, *13*, 651. https://doi.org/10.3390/diagnostics13040651

Academic Editors: Semen A. Kurkin and Alexander E. Hramov

Received: 19 December 2022
Revised: 4 February 2023
Accepted: 7 February 2023
Published: 9 February 2023

1. Introduction

1.1. Preamble

Primary brain tumors refer to a heterogeneous group of tumors developing from various types of cells within the Central Nervous System (CNS) [1]. Primary brain tumors can be benign (non-malignant) or malignant. Non-malignant tumors grow slowly and do not spread to neighboring tissues, while malignant tumors expand rapidly and tend to invade other tissues [2]. According to the revised 2021 World Health Organization (WHO) classification of CNS tumors [3], brain tumors are mainly classified as: (a) Gliomas, glioneuronal tumors, and neuronal tumors; (b) Choroid plexus tumors; (c) Embryonal tumors; (d) Pineal tumors; (e) Cranial and paraspinal nerve tumors; (f) Meningiomas;

(g) Mesenchymal, non-meningothelial tumors; (h) Melanocytic tumors; (i) Hematolymphoid tumors; (j) Germ cell tumors; (k) Tumors of the sellar region; and (l) metastases to the CNS. The Central Brain Tumor Registry of the United States (CBTRUS) reported that about 28.3% of all brain and other CNS tumors diagnosed in the United States between 2015 and 2019 were malignant, while 71.7% were non-malignant [4]. It is also reported that gliomas, including glioblastoma, pilocytic astrocytoma, oligodendroglioma, ependymoma, and a few rare histopathologies, were the most common malignant primary brain tumor type. Specifically, in the report, it is indicated that the most commonly occurring tumor was glioblastoma, which accounted for 14.2% of all tumors and 50.1% of all malignant tumors. According to the pathological and genetic properties, on the other hand, gliomas are classified by the WHO into four grades. Grades I and II are categorized as low-grade gliomas (LGG) such as pilocytic astrocytomas (WHO I), and diffuse low grade gliomas including diffuse astrocytomas and oligodendrogliomas (WHO II). Grades III and IV are categorized as high-grade gliomas (HGG), including anaplastic astrocytomas and anaplastic oligodendrogliomas (WHO III), and glioblastoma (WHO IV) [3,5]. Obviously, when compared to HGG, LGG are mostly considered as less-threatening tumors [6].

Diagnosis of gliomas at early stages is strictly necessary for a patients' survival. For diagnosing tumors, the radiologists use different types of procedures or technologies such as biopsy, cerebrospinal fluid analysis, and medical imaging. However, with the rapid developments of computer science, imaging technologies have become more popular due to their accuracy and low risk to patients. Today, three imaging technologies such as X-rays, computed tomography (CT), positron emission tomography (PET), and magnetic resonance imaging (MRI), are commonly used in the diagnosis of brain tumors. Among them, MRI is the most preferred imaging technology because of its high-resolution image data that are useful for the detection and classification of tumors. However, the identification of tumors from MRI data is not an easy task for the experts, as they need to put forth significant effort in order to provide a diagnosis in a limited time. Therefore, automated diagnostic systems based on machine learning (ML) approaches have started to be used [7]. ML is mainly aimed at classifying tumors into the specific classes, such as tumor substructure, tumor/non-tumor, or benign/malignant tumor.

Various ML methods based on particular classifiers such as Support Vector Machine (SVM) [8–10], Extreme Learning Machine-Improved Particle Swarm Optimization (ELM-IPSO) [11], K-Nearest Neighbors (KNN) [12], Artificial Neural Networks (ANN) [13], and Random Forest (RF) [14,15] are used for glioma detection in the literature. However, it is worth noting that such traditional ML methods apply pre-designed feature extraction from the MRI data. Evidently, pre-selection of features might adversely affect the performance of these traditional ML methods [16]. In order to resolve this concern, deep learning (DL) approaches have gained increased attention from many researchers [17]. One of the most important advantages over traditional ML methods is that hand-crafted features are not required in the implementation of DL approaches. Even complicated patterns can be discriminated by automatic feature learning. This enables practitioners to generate faster and better insights from large MRI datasets.

In addition to glioma detection, the use of DL approaches is also very popular in real-life applications in different fields, such as groundwater level prediction [18], nonlinear distributed thermal processes [19], or even the aquaculture and fishery industry [20], due to its easier implementation and higher computational efficiency. Many studies have been conducted to propose DL approaches to be implemented for efficient detection and classification of brain tumors as glioma, meningioma, or pituitary tumors from MRI data [21–33]. DL approaches based on Convolutional Neural Networks (CNNs) have attracted much interest, as they require minimum preprocessing [34–40]. However, it is very important to understand which CNN architecture would work well for detection and classification of brain tumors under various conditions including the development environment or programming aspects. Until now, two high-level programming environments, namely MATLAB and Python, have been commonly preferred in the literature to implement the DL

approaches for the detection of brain tumors. Therefore, it is necessary to investigate the impact of these programming environments on the accuracy of CNN-based brain tumor detection.

1.2. Related Works and Research Gaps

High-level programming environments such as MATLAB, Python, R, Scala, Julia, or Java are able to empower scientist and engineers in order to implement their ideas in a faster way. This is due to the fact that the computational difficulties of low-level programming languages, such as C, C++, or FORTRAN, can be relaxed by high-level programming environments, which allow users to deal with mathematically heavy problems by means of minimal programming. Among high-level programming environments, the use of MATLAB and Python is mostly preferred in DL implementations due to their easy learning, simplicity of use, and adaptability features.

The performances of MATLAB and Python compared to each other has been a subject of debate over the years. In this context, the performance of MATLAB has been comparatively assessed with Python for different applications in various fields [41–46]. In [41], the aim was to offer a well-structured teaching language that enabled engineering students to quickly express their algorithms. In [42], the comparisons were made in terms of execution times for a macroeconomic application. The study presented in [43] was conducted for researchers in the field of economics in order to offer a best programming environment suited to their own purpose. In [44], the performance of the programming environments in a bioinformatics application were compared in terms of their memory requirement and ease of integration. In [45], the runtimes of the programming environments were compared for complex number calculations used in electromagnetic applications. In [46], accuracies of the programming environments for the computation of a robotic arm end effector matrix were compared.

Motivated by the aforementioned discussion, there have been no studies presented that focus on the efficiency of MATLAB and Python on CNN-based brain tumor detection from MRI data. This, in fact, indicates a significant research gap that might be critical for many practitioners in their appropriate implementation of DL approaches for brain tumor detection.

1.3. Purpose and Contributions

This research work aims to investigate the impact of two major programming environments, namely MATLAB and Python, on the accuracy of CNN-based brain tumor detection, particularly glioma, from a well-known MRI dataset. For this purpose, extensive experiments on Brain Tumor Segmentation (BraTS) dataset are performed by implementing two popular and simple CNN architectures: a standard three-dimensional (3D) U-Net [47] and V-Net [48] in MATLAB and Python. In the experiments, due to its larger size and higher resolution, the BraTS 2016 and 2017 dataset, consisting of multiparametric-magnetic MRI from patients diagnosed with either HGG or LGG, was used [49]. Experimental results show that using Python with Google Colaboratory (Colab) offers significant advantages in terms of training time and accuracy over MATLAB. Results also indicate that the 3D U-Net model has higher accuracy in comparison to the V-Net model on the used dataset. Thus, it is believed that the comparative study provided in this paper would attract the research community working in the area of brain tumor detection from MRI images, and help them to explore some future research directions as well as applicability of the models to BraTS dataset. The main contributions of this study can be summarized as follows:

(a) This is the first study that comparatively assesses the impact of two major programming environments on the accuracy of CNN-based glioma detection.
(b) Glioma detection performances of two popular CNN architectures, namely 3D U-Net and V-Net, are compared using the BraTS dataset (2016 and 2017) for the first time in the literature.

The rest of the article is organized as follows. In Section 2, an overview of MATLAB and Python for DL implementations is provided. Then, in Section 3, the details of the experiments performed within the context of this study are presented. This is followed by Section 4, where experimental results are presented. Further discussions and future research directions are addressed in Sections 5 and 6, respectively. Finally, the article is concluded in Section 7.

2. Overview of MATLAB and Python for Deep Learning

In this section, a brief discussion on the use of two major programming environments in DL implementations is presented to provide a better understanding before describing the experiments and results.

It is widely known that DL is a branch of ML, which teaches computers to learn from experience. In DL, neural networks combining nonlinear processing layers are used to learn useful features from the data. The effectiveness of DL models in object classification has been thoroughly discussed in the literature. However, the efficiency of MATLAB and Python in DL model implementations is still under debate by the research communities.

MATLAB is a popular high-level programming language often employed both in industry and research for numerical computations [50]. It is very mature and widespread among the scientific community as well as practitioners, with its user-friendly toolboxes and libraries for DL implementations. It supports Open Neural Network Exchange (ONNX), which is an open standard that defines common file format and set of operators to represent DL models in various frameworks. It is interoperable with Python, which empowers researchers to use MATLAB and Python together. Moreover, it is easy to preprocess datasets with domain-specific applications for various types of data. In other words, it is possible to check and fix problems in order to build or modify complex network architectures for transfer learning before the training process. It is also possible to produce code supporting optimized libraries such as the ARM Compute Library, NVIDIA TensorRT, and Intel Math Kernel Library for Deep Neural Networks (MKL-DNN). This multi-program deployment feature significantly improves its performance. Furthermore, MATLAB Deep Learning Toolbox provides a framework in order to design and implement DL networks or models. For instance, Long Short-Term Memory (LSTM) networks and CNNs can be easily applied by the users. Its applications enable users to update the network architectures, to check the progress in preparation, to visualize the layer activations, and to monitor training progress graphically. For training on a modest dataset, it is possible to operate transfer learning through the pre-trained network models or models trained with libraries such as Keras, Caffe, and TensorFlow. For training on larger datasets, the process can be boosted on a single or multiple graphics processing units (GPUs) with Parallel Computing Toolbox. In addition, the training process can be scaled up to clusters and clouds, such as Amazon Elastic Compute Cloud (EC2) GPU instances, and NVDIA GPU Cloud with MATLAB Parallel Server. Although MATLAB has many advantages for DL, not being open source might be considered as an important drawback. MATLAB is proprietary software, and it is expensive because of its commercial license. This, in fact, limits the flexibility of a professional programmer. In addition, the use of NVDIA GPU Cloud and Amazon EC2 instances is not free, which can be considered as another drawback in the training of larger datasets.

Python, on the other hand, is another high-level language that is increasingly used in various sectors [51]. It is open-source, and it has a rich ecosystem from which a variety of packages and libraries can be downloaded and installed for different tasks and purposes. It has very easy syntax and commands; therefore, it appeals to many programmers as it is very easy to learn. This makes it easier to build models for DL implementations. Since it is a general-purpose language, a set of complex DL tasks can be accomplished and prototypes can be easily built in order to test a product for DL purposes. It has an extensive set of libraries that are widely used for DL, such as PyTorch, Keras, Tensorflow, Caffe, and others. It also offers a variety of visualization tools, namely Matplotlib, seaborn and gplot, for better

understanding of data. It could be a good choice for DL, as it offers flexible programming which enables the programmers to select the programming styles, including imperative style, functional style, object-oriented style, and procedural style. Moreover, Google Colab can be used to execute arbitrary python code for DL implementations. Colab provides free access to utilize computing resources including GPUs. Hence, when DL implementation issues such as the storage of dataset and GPU in DL implementations are concerned, Python enables users to use free high-performance GPUs and open cloud environments. This can be considered as an important advantage for training large datasets.

It is surely beyond doubt that the interest in the use of Python for DL is increasing day by day, whereas the benefits provided by MATLAB should not be ignored. However, it is very important to note that deciding on the right programming environment for DL implementations depends on the user's perceptions and expectations. Thus, it might be better to use MATLAB and Python interactively for utilizing the abilities of each environment. We believe that this hybrid approach could be expected to be more widespread among the researchers in the near future.

3. Experiments

3.1. Dataset

Several open source datasets such as BraTS [52], Harvard [53], RIDER (Reference Database to Evaluate Response) [54], and ISLES (Ischemic Stroke Lesion Segmentation) [55] have been used in brain tumor analysis. However, the BraTS datasets, which are an open source repository of multi-institutional radiological data, are mostly used by the researchers in ML competitions (challenges) in order to optimize or evaluate their proposed models each year, due to the fact that BraTS datasets are the most challenging MRI datasets [56].

The brain tumor dataset utilized in the experiments is a volumetric medical image dataset, which is a subset of the data (Task 1) used in the 2016 and 2017 BraTS challenges [57,58]. This dataset was selected to be used in the experiments, as it is created for the challenge of locating the complex and heterogeneously-located targets while other BraTS datasets are created for different tasks [49]. The dataset consists of 484 multimodal multisite MRI images from patients diagnosed with either HGG or LGG. Each sample has four modalities, such as FLAIR (Fluid-Attenuated Inversion Recovery), T1w (T1-weighted), T1gd (T1-weighted with gadolinium contrast enhancement), T2w (T2-weighted), with ground truth labels, namely background, enhancing and non-enhancing tumor, and edema segmentations. Each image has $240 \times 240 \times 155$ volume size, and has uniform, 1 mm^3, voxel resolution. A sample of the dataset before preprocessing is shown in Figure 1 (adopted from [59]).

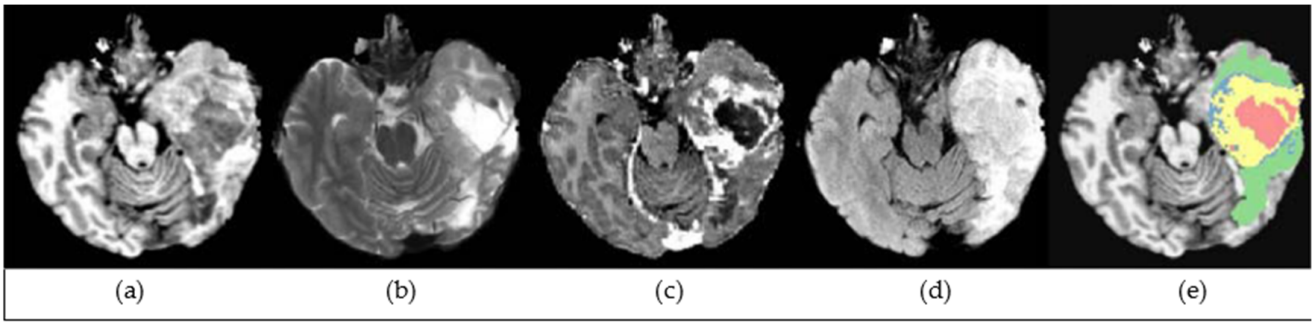


Figure 1. A sample of the dataset: (**a**) T1w (T1-weighted); (**b**) T2w (T2-weighted); (**c**) T1gd (T1-weighted with gadolinium contrast enhancement); (**d**) FLAIR (Fluid Attenuated Inversion Recovery); (**e**) Ground truth.

3.2. Deep Learning Approaches Used in the Experiments

In the literature, various DL approaches based on CNN architectures such as 3DConvNet [60], ResNet [61], DenseNet [62], GoogleNet [63], AlexNet [37], U-Net [64], V-Net [65] and VGG16 [66] have been used in MRI-based glioma detection, classification, segmentation, and grading. In general, all of these architectures demonstrate a good performance. However, in this study, a 3D version of the original U-Net architecture, which is adopted from [47], and a standard V-Net architecture [44] were chosen to be used in the experiments. One of the reasons is that they comply with our inclusion criteria, which included simplicity, relevance with glioma detection, and applicability to a volumetric medical image dataset. It should be noted that although the 3D U-Net and V-Net have been already applied to similar or closely-relevant applications, their glioma detection performances have not been comparatively assessed on the BraTS dataset (2016 and 2017) yet. This, in fact, is the other reason for choosing the 3D U-Net and V-Net architectures to be used in the experiments. In the following subsections, these architectures are summarized in technical aspects.

3.2.1. 3D U-Net

A standard (two-dimensional: 2D) U-Net architecture is mainly composed of three parts: (a) Encoder, (b) bottleneck, (c) decoder [67]. In the encoder part, the prospective region of the image is identified. There are 3×3 convolution layers with activation functions, and 2×2 max pooling layers where the feature maps are doubled. The bottleneck part of the architecture is a path that assembles the encoding and decoding path. The decoder part has transposed convolutional layers with concatenation, and 3×3 convolution layers with activation functions.

For glioma segmentation and/or detection from MRI data, the use of DL approaches based on U-Net variants has been proposed in many works [47,64,68–78]. However, since a volumetric image dataset is used in this study, a standard U-Net architecture and its variants developed for 2D datasets become inapplicable. For a proper implementation, it needs to be advanced by changing all 2D operations with their 3D counterparts. Hence, basic 3D U-Net [47] and its many variants, such as residual symmetric 3D U-Net [79], 3D U-Net++ [80], Attention 3D U-Net [81], and Separable 3D U-Net (S3D-UNet) [82] have been proposed for brain tumor segmentation. In this study, basic 3D U-Net framework is adapted due to its simplicity.

The network architecture of basic 3D U-Net is shown in Figure 2. It should be noted that the numbers shown in the red boxes correspond to the number of filters. As shown in the figure, the input layer is followed by a convolutional layer, batch normalization, and a Rectified Linear Unit (ReLU) activation layer. Through the layers, padding is set to same, and the kernel size is kept constant at, $3 \times 3 \times 3$. In addition, the max pooling layer is sized at $2 \times 2 \times 2$ in the encoding path. In the decoding path, on the other hand, the number of filters is reduced by half after the transposed convolutional layers is applied with stride 2. A convolutional layer with a sigmoid activation function is then used for the segmentation of glioma.

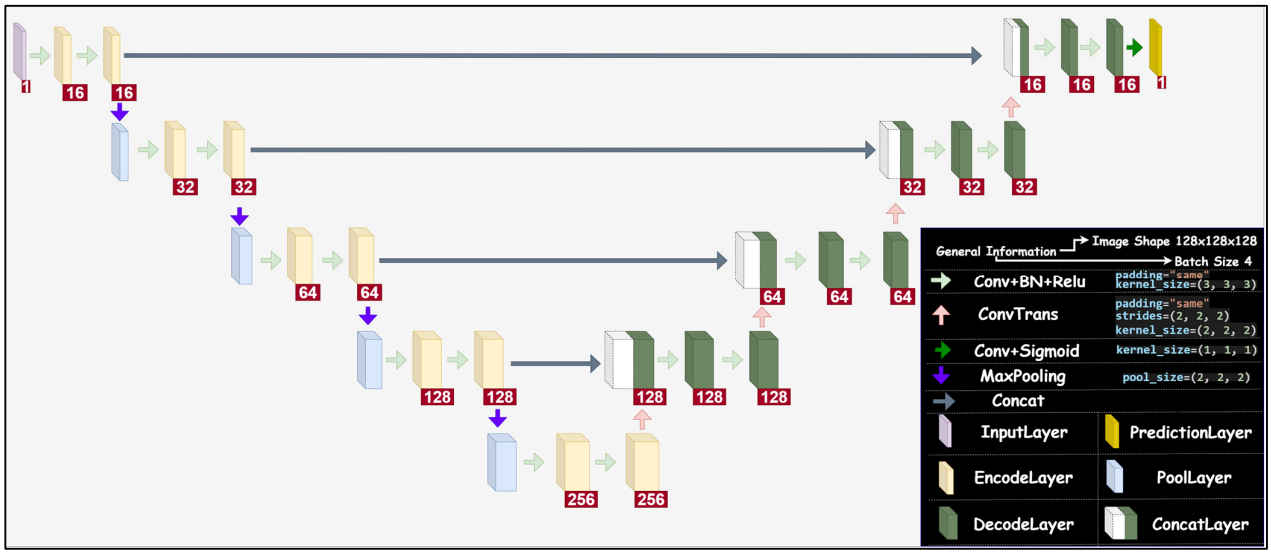


Figure 2. The network architecture of 3D U-Net used in this study.

3.2.2. V-Net

The standard V-Net architecture is a 3D CNN that is a modified version of standard U-Net for volumetric medical image segmentation [48]. In order to enhance the efficiency, various V-Net variants have been presented for brain tumor segmentation, such as Deep Supervised 3D Squeeze-and-Excitation V-Net (DSSE-V-Net) [65], Cascaded V-Net [83,84], Attention V-Net [85], and 3D AGSE-VNet [86]. For the sake of simplicity, the standard U-Net architecture is utilized in this study.

In comparison to the 3D U-Net architecture, in a standard U-Net architecture, max pooling operations are replaced with convolutional layers which have a $5 \times 5 \times 5$ kernel size in each stage. Moreover, throughout the network, Parametric ReLU (PReLu) non-linearities are applied. Another difference in comparison to U-Net is that the features extracted from the early stages of the left part of the network are forwarded to the right part of the network in order to gather fine-grained detail.

The network architecture of a standard V-Net is shown in Figure 3. In the left part of the network, there are four stages along with an input stage. The data resolution is reduced by means of convolutions with $2 \times 2 \times 2$ kernels applied with stride 2. In the right part of the network, on the other hand, the very last convolutional layer has $1 \times 1 \times 1$ kernel size, and produces outputs that are the same size as the input volume. After each stage, de(down)-convolution operation is applied, which is followed by convolutional layers with $1 \times 1 \times 1$ strides.

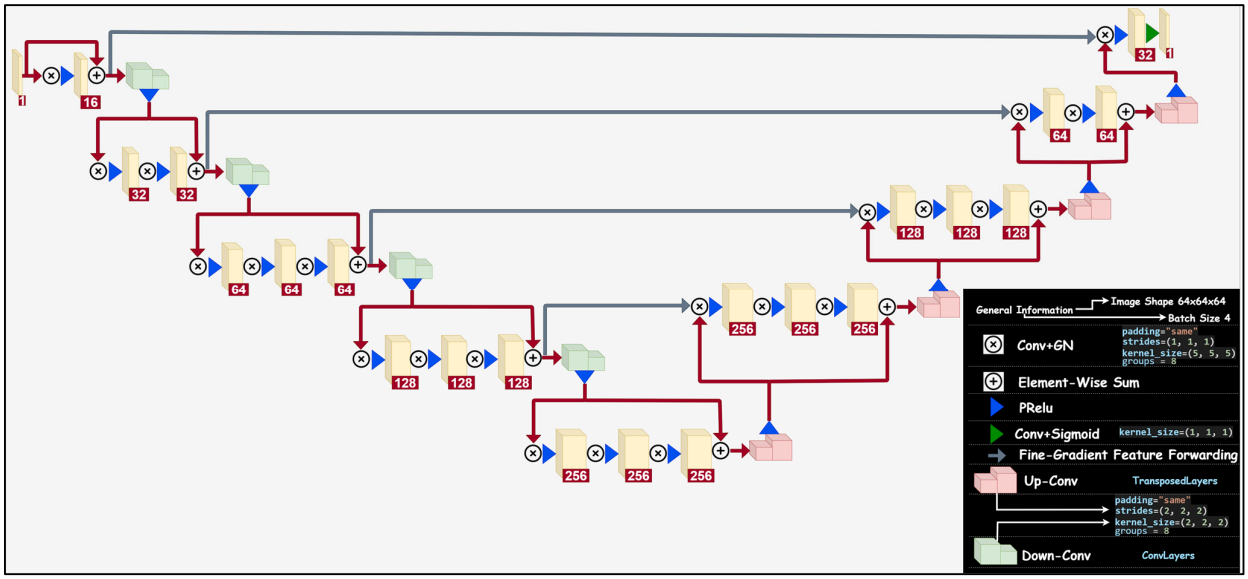


Figure 3. The network architecture of V-Net used in this study.

3.3. Preprocessing and Implementation Details

Before feeding the data into the U-Net and V-Net models, the data were preprocessed considering the networks' architectures. In this context, the input dataset was resized to 128 × 128 × 128 dimensions for the 3D U-Net model, while it was resized to 64 × 64 × 64 dimensions for the V-Net model. The images in both datasets were then randomly flipped to reduce overfitting in the training of the models. Next, the dataset for each model is split up into the training set (400 images), the test set (55 images) and the validation set (29 images).

The 3D U-Net and V-Net models described in Section 3.1 were implemented in Python and MATLAB. Google Colaboratory (Colab) was used as a development platform in order to implement the models in Python. In MATLAB implementations, on the other hand, the workstation equipped with NVIDIA Quadro RTX 5000 was used. Both 3D U-Net and V-Net models were trained on the same dataset introduced in the previous sub-section. In order to train the models, the Adam optimizer was used due to its faster computation time, and because it requires fewer parameters for tuning than other optimizers such as Gradient Descent, Adagrad, RMSProp, or AdaDelta. In addition, a smaller learning rate was decided to be used, which was set to 0.0003. Since it requires more training epochs in order to properly converge to a suboptimal solution, the epoch was set to 50. Moreover, in order to achieve the best performance of the workstation in terms of its computational power efficiency to process all images in parallel, batch size was varied and then determined to be set to 4.

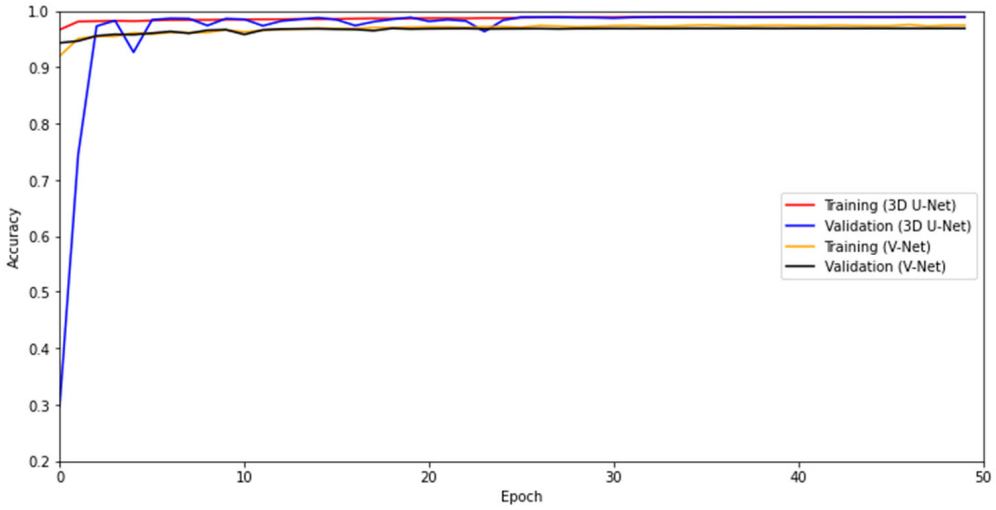
It is also worth noting that the input patch size was 4 × 128 × 128 × 128 to implement the 3D U-Net model, while it was 4 × 64 × 64 × 64 for the V-Net model, since there are four MRI modalities in the original dataset.

4. Results of the Experiments

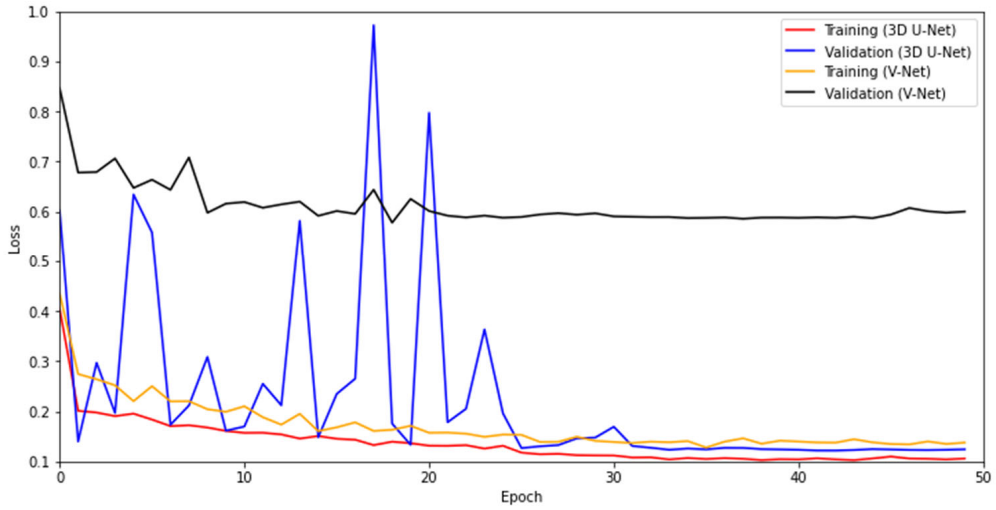
Generally, the effectiveness of ML algorithms is evaluated under different goals, such as easy training, long lifetime, good performance, and rapid production. In order to compare the effectiveness of ML models or algorithms, development-based and production-based approaches can be used. While development-based approaches include statistical tests, loss functions and metrics, and learning curves (training and validation learning curves), production-based approaches include two basic parameters: time complexity and space complexity.

On the other hand, since the main target of this study is to investigate the impact of two major programming environments on the accuracy of DL-based glioma detection, basic comparable parameters, such as learning curves, loss (accuracy) metrics, and training time, were chosen to evaluate the experimental results. Other comparable parameters can still be used; however, they are more applicable for comparing the effectiveness of DL algorithms comprehensively. Thus, firstly, the models were validated on the dataset, and their accuracies were recorded. Then, based on the comparative performances of the models, the effects of the programming environments on the results are assessed.

The training and validation accuracy and loss of the 3D U-Net and V-Net models implemented in Python are shown in Figures 4a and 4b, respectively. The 3D U-Net model achieves a training accuracy of 98.7% with a training loss of 13.7% and a validation accuracy of 96.6% with a validation loss of 36.2%. The test accuracy, moreover, is found to be 98.9% while the test loss is found to be 21.9%. On the other hand, the V-Net model provides a training accuracy of 96.9% with a training loss of 17.1% and a validation accuracy of 96.6% with a validation loss of 61.3%. Furthermore, the test accuracy of the model is found to be 96.9%, while the test loss is found to be 40.6%.



(**a**)



(**b**)

Figure 4. For the 3D U-Net and V-Net models implemented in Python: (**a**) Training and validation accuracy, (**b**) Training and validation loss.

As for the 3D U-Net model implemented in MATLAB, the model achieves a training accuracy of 98.6% with a training loss of 26.4% and a validation accuracy of 97.7% with a validation loss of 37.6%. The test accuracy achieved by the model is 97.8%, and the test loss is 26.7%. The V-Net model implemented in MATLAB, on the other hand, achieves a training accuracy of 97.7 % with a training loss of 25.4% and a validation accuracy of 97.5% with a validation loss of 37.9%. Furthermore, the V-Net model achieves test accuracy of 97.2% while the test loss of 43.2%.

The comparative performances of the 3D U-Net and V-Net models implemented both in Python and MATLAB are summarized in Table 1. It is clear that the 3D U-Net model has higher accuracy when compared to the V-Net model regardless of the programming environment used in the experiments. Particularly, low training loss (17.1%) but high test loss (40.6%) obtained in Python experiments show that the V-Net model could be overfitting. This suggests that the V-Net model might not be applicable to the used BraTS dataset for glioma detection. Therefore, in order to assess the impact of programming environments on the detection accuracies, only the results achieved for 3D U-Net model are considered.

Table 1. Comparison of the models implemented in Python and MATLAB.

Metric	Python		MATLAB	
	3D U-Net	**V-Net**	**3D U-Net**	**V-Net**
Training Accuracy (%)	98.7	96.9	98.6	97.7
Training Loss (%)	13.7	17.1	26.4	25.4
Validation Accuracy (%)	96.6	96.6	97.7	97.5
Validation Loss (%)	36.2	61.3	37.6	37.9
Test Accuracy (%)	98.9	96.9	97.8	97.2
Test Loss (%)	21.9	40.6	26.7	43.2
Training Time (hr.)	~4	~4	~38	~38

From the results listed in Table 1, it can be seen that the accuracy of the 3D U-Net model is highly affected by the programming environment used in the experiments. The test accuracy of 98.9% achieved from Python is reduced to the accuracy of 97.8% when MATLAB is used. This is also the case for training loss of the models, which decreases from 26.4%, obtained in MATLAB, to 13.7%, obtained in Python.

Concerning the training time of the models, on the other hand, it took about 4 hours for training both the 3D U-Net and V-Net model in Python, while it took about 38 hours in MATLAB. The reason is that a free GPU supported and provided by Google Colab is much faster than the GPU of the workstation used to perform experiments in MATLAB. Moreover, during the implementation of models in Python, the data is retrieved from a free cloud service of Google Colab, whereas it is retrieved from MATLAB folders. Therefore, using Python with Google Colab offers a significant advantage over MATLAB in terms of training time.

5. Discussion

The results achieved from the experiments can be discussed under two main heads. One is the effect of programming environments on the accuracy of CNN-based models in glioma detection from MRI images. From the experimental results, it can be concluded that the detection accuracies of the models can be affected by the used programming environment. This can be clearly seen in Table 1, where the test accuracy of 3D U-Net obtained from Python is decreased by 1.1% when MATLAB is used. This is also the case for test loss, which is increased by 4.8% when MATLAB is used. On the other hand, when the training time is considered, Python has an important advantage over MATLAB due to Google Colab, with the native features available for any user such as faster GPUs and cloud data storage. In Table 1, it is clear that the training time takes longer when MATLAB

is used. However, it is still possible to achieve higher training times with MATLAB, if a powerful workstation is available.

The second head is the comparison of the effectiveness of 3D U-Net and V-Net models utilized for detecting glioma images (tumor and non-tumor) from BraTS 2016 and 2017 dataset. The experimental results show that the 3D U-Net model has higher accuracy than the V-Net model. However, its performance needs to be compared with the state-of-the-art in order to quantify its efficiency over other available models. In the literature, only a few studies that test DL based on the same dataset for detecting glioma images (tumor and non-tumor) [87,88]. In [87], the SoftMax layer is used with the central clustering algorithm for feature extraction. The extracted features are then used in a CNN-based algorithm. Results show that the proposed approach provides 96% accuracy. In [88], a data augmentation-based DL approach is proposed. By means of a ResNet-50 classifier, the experimental results show that the proposed approach achieves 91.08% accuracy. Therefore, the results obtained in the study offer that the 3D U-Net with 98.9% test accuracy has an ability to reach higher accuracies in glioma detection when compared to the approaches proposed in both [87] and [88].

6. Future Research Directions

Although this study addresses some major issues on DL developments, there are still important research opportunities. In fact, the authors are currently working on some specific aspects. First of all, some other programming environments have gained interest in recent years, such as R, Scala, Julia, and Java, and are planned to be used in DL implementations in order to obtain more comprehensive comparison results. In this way, it will be possible to provide a deeper insight regarding the programming environments to be used for implementing DL-based models in brain tumor detection.

In this study, among the CNN-based models, 3D U-Net and V-Net were chosen to be implemented in MATLAB and Python for brain tumor detection. Obviously, other well-known CNN-based models could have been used in the study. In the literature, however, glioma detection performances of 3D U-Net and V-Net on BraTS dataset (2016 and 2017) have not been compared yet. Therefore, the main motivation behind our choice was to compare their glioma detection performance for the first time in the literature. In this context, our second task is to conduct new experiments on the most up-to-date BraTS datasets by accounting for other well-known CNN-based models, in order to evaluate their glioma detection accuracies.

It is important to note that the objective of the detection process in this study is to identify whether glioma is present in MRI images or not. As mentioned earlier, the primary brain tumors are categorized as meningioma, pituitary, and glioma. Many studies presented in the literature focus on the identification or classification of the primary brain tumors from MRIs. Moreover, classifying glioma grades as LGG or HGG from MRI images has received great interest in the literature. Thus, as a third task, the current study will be directed toward examining the effects of programming environments on the classification of primary brain tumors or glioma grades.

Furthermore, it would be interesting to evaluate the experimental results from the clinician's point of view, due to the fact that clinicians mostly need accurate and reliable results. This idea has therefore motivated us to involve clinicians in our future works.

7. Conclusions

This work was devoted to examining the impact of two major programming environments, namely MATLAB and Python, on the accuracy of glioma detection using CNN-based DL models. For this purpose, experiments were performed in which two popular CNN architectures were implemented on a well-known BraTS dataset (2016 and 2017). According to the experimental results, the use of Python with Google Colab brings some benefits in comparison to MATLAB. More specifically, using Python might affect the accuracy of the models positively, and require less training time. On the other hand, when compared to

the V-Net model, the 3D U-Net model demonstrates higher accuracy. This suggests that the V-Net model could not be applicable to BraTS datasets, and further modifications might be required to improve its accuracy.

In general, it is expected that the results obtained in this study might offer some insights on the most appropriate DL development environment for brain tumor detection. In the near future, the authors are planning to enrich the presented study by considering different datasets, DL models, and programming environments in order to provide more comprehensive results.

Author Contributions: Conceptualization, investigation, methodology, software, data curation, V.S.Y. and M.A.; validation, formal analysis, Y.D., R.O.D., A.K. and A.S.; writing—original draft preparation, V.S.Y. and Y.D.; writing—review and editing, supervision, R.O.D., A.K. and A.S.; All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by Atilim University Undergraduate Research Projects: [Grant Number ATU-LAP-2021-05].

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are openly available at http://medicaldecathlon.com/, accessed on 19 December 2022.

Acknowledgments: Authors would like to thank students of Atilim University for their voluntary contribution to this study.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Lapointe, S.; Perry, A.; Butowski, N.A. Primary Brain Tumours in Adults. *Lancet* **2018**, *392*, 432–446. [CrossRef]
2. DeAngelis, L.M. Brain Tumors. *N. Engl. J. Med.* **2001**, *344*, 114–123.
3. Louis, D.N.; Perry, A.; Wesseling, P.; Brat, D.J.; Cree, I.A.; Figarella-Branger, D.; Hawkins, C.; Ng, H.K.; Pfister, S.M.; Reifenberger, G.; et al. The 2021 WHO Classification of Tumors of the Central Nervous System: A Summary. *Neuro-Oncol.* **2021**, *23*, 1231–1251. [CrossRef]
4. Ostrom, Q.T.; Price, M.; Neff, C.; Cioffi, G.; Waite, K.A.; Kruchko, C.; Barnholtz-Sloan, J.S. CBTRUS Statistical Report: Primary Brain and Other Central Nervous System Tumors Diagnosed in the United States in 2015–2019. *Neuro-Oncol.* **2022**, *24*, v1–v95.
5. Alis, D.; Bagcilar, O.; Senli, Y.D.; Isler, C.; Yergin, M.; Kocer, N.; Islak, C.; Kizilkilic, O. The Diagnostic Value of Quantitative Texture Analysis of Conventional MRI Sequences Using Artificial Neural Networks in Grading Gliomas. *Clin. Radiol.* **2020**, *75*, 351–357. [CrossRef]
6. Bauer, S.; Wiest, R.; Nolte, L.-P.; Reyes, M. A Survey of MRI-Based Medical Image Analysis for Brain Tumor Studies. *Phys. Med. Biol.* **2013**, *58*, R97. [CrossRef]
7. Amin, J.; Sharif, M.; Haldorai, A.; Yasmin, M.; Nayak, R.S. Brain Tumor Detection and Classification Using Machine Learning: A Comprehensive Survey. *Complex Intell. Syst.* **2021**, *8*, 3161–3183. [CrossRef]
8. Zöllner, F.G.; Emblem, K.E.; Schad, L.R. SVM-Based Glioma Grading: Optimization by Feature Reduction Analysis. *Z. Für Med. Phys.* **2012**, *22*, 205–214. [CrossRef]
9. Javed, U.; Riaz, M.M.; Ghafoor, A.; Cheema, T.A. MRI Brain Classification Using Texture Features, Fuzzy Weighting and Support Vector Machine. *Prog. Electromagn. Res. B* **2013**, *53*, 73–88. [CrossRef]
10. Lahmiri, S. Glioma Detection Based on Multi-Fractal Features of Segmented Brain MRI by Particle Swarm Optimization Techniques. *Biomed. Signal Process. Control* **2017**, *31*, 148–155.
11. Nachimuthu, D.S.; Baladhandapani, A. Multidimensional Texture Characterization: On Analysis for Brain Tumor Tissues Using MRS and MRI. *J. Digit. Imaging* **2014**, *27*, 496–506. [CrossRef]
12. Amin, J.; Sharif, M.; Raza, M.; Saba, T.; Anjum, M.A. Brain Tumor Detection Using Statistical and Machine Learning Method. *Comput. Methods Programs Biomed.* **2019**, *177*, 69–79. [CrossRef]
13. Kickingereder, P.; Isensee, F.; Tursunova, I.; Petersen, J.; Neuberger, U.; Bonekamp, D.; Brugnara, G.; Schell, M.; Kessler, T.; Foltyn, M. Automated Quantitative Tumour Response Assessment of MRI in Neuro-Oncology with Artificial Neural Networks: A Multicentre, Retrospective Study. *Lancet Oncol.* **2019**, *20*, 728–740.
14. Pinto, A.; Pereira, S.; Dinis, H.; Silva, C.A.; Rasteiro, D.M. Random Decision Forests for Automatic Brain Tumor Segmentation on Multi-Modal MRI Images. In Proceedings of the 2015 IEEE 4th Portuguese meeting on bioengineering (ENBENG), Porto, Portugal, 26–28 February 2015; IEEE: Piscataway, NJ, USA, 2015; pp. 1–5.

15. Abbasi, S.; Tajeripour, F. Detection of Brain Tumor in 3D MRI Images Using Local Binary Patterns and Histogram Orientation Gradient. *Neurocomputing* **2017**, *219*, 526–535.
16. Shaver, M.M.; Kohanteb, P.A.; Chiou, C.; Bardis, M.D.; Chantaduly, C.; Bota, D.; Filippi, C.G.; Weinberg, B.; Grinband, J.; Chow, D.S. Optimizing Neuro-Oncology Imaging: A Review of Deep Learning Approaches for Glioma Imaging. *Cancers* **2019**, *11*, 829. [CrossRef]
17. Nazir, M.; Shakil, S.; Khurshid, K. Role of Deep Learning in Brain Tumor Detection and Classification (2015 to 2020): A Review. *Comput. Med. Imaging Graph.* **2021**, *91*, 101940.
18. Afan, H.A.; Ibrahem Ahmed Osman, A.; Essam, Y.; Ahmed, A.N.; Huang, Y.F.; Kisi, O.; Sherif, M.; Sefelnasr, A.; Chau, K.; El-Shafie, A. Modeling the Fluctuations of Groundwater Level by Employing Ensemble Deep Learning Techniques. *Eng. Appl. Comput. Fluid Mech.* **2021**, *15*, 1420–1439. [CrossRef]
19. Fan, Y.; Xu, K.; Wu, H.; Zheng, Y.; Tao, B. Spatiotemporal Modeling for Nonlinear Distributed Thermal Processes Based on KL Decomposition, MLP and LSTM Network. *IEEE Access* **2020**, *8*, 25111–25121. [CrossRef]
20. Banan, A.; Nasiri, A.; Taheri-Garavand, A. Deep Learning-Based Appearance Features Extraction for Automated Carp Species Identification. *Aquac. Eng.* **2020**, *89*, 102053.
21. Díaz-Pernas, F.J.; Martínez-Zarzuela, M.; Antón-Rodríguez, M.; González-Ortega, D. A Deep Learning Approach for Brain Tumor Classification and Segmentation Using a Multiscale Convolutional Neural Network. *Healthcare* **2021**, *9*, 153.
22. Haq, E.U.; Jianjun, H.; Li, K.; Haq, H.U.; Zhang, T. An MRI-Based Deep Learning Approach for Efficient Classification of Brain Tumors. *J. Ambient Intell. Humaniz. Comput.* **2021**, 1–22. [CrossRef]
23. Raza, A.; Ayub, H.; Khan, J.A.; Ahmad, I.; Salama, A.S.; Daradkeh, Y.I.; Javeed, D.; Ur Rehman, A.; Hamam, H. A Hybrid Deep Learning-Based Approach for Brain Tumor Classification. *Electronics* **2022**, *11*, 1146. [CrossRef]
24. Zahoor, M.M.; Qureshi, S.A.; Bibi, S.; Khan, S.H.; Khan, A.; Ghafoor, U.; Bhutta, M.R. A New Deep Hybrid Boosted and Ensemble Learning-Based Brain Tumor Analysis Using MRI. *Sensors* **2022**, *22*, 2726.
25. Khan, M.S.I.; Rahman, A.; Debnath, T.; Karim, M.R.; Nasir, M.K.; Band, S.S.; Mosavi, A.; Dehzangi, I. Accurate Brain Tumor Detection Using Deep Convolutional Neural Network. *Comput. Struct. Biotechnol. J.* **2022**, *20*, 4733–4745. [CrossRef]
26. Ullah, N.; Khan, J.A.; Khan, M.S.; Khan, W.; Hassan, I.; Obayya, M.; Negm, N.; Salama, A.S. An Effective Approach to Detect and Identify Brain Tumors Using Transfer Learning. *Appl. Sci.* **2022**, *12*, 5645.
27. Gupta, R.K.; Bharti, S.; Kunhare, N.; Sahu, Y.; Pathik, N. Brain Tumor Detection and Classification Using Cycle Generative Adversarial Networks. *Interdiscip. Sci. Comput. Life Sci.* **2022**, *14*, 485–502.
28. Sekhar, A.; Biswas, S.; Hazra, R.; Sunaniya, A.K.; Mukherjee, A.; Yang, L. Brain Tumor Classification Using Fine-Tuned GoogLeNet Features and Machine Learning Algorithms: IoMT Enabled CAD System. *IEEE J. Biomed. Health Inform.* **2021**, *26*, 983–991.
29. Samee, N.A.; Mahmoud, N.F.; Atteia, G.; Abdallah, H.A.; Alabdulhafith, M.; Al-Gaashani, M.S.; Ahmad, S.; Muthanna, M.S.A. Classification Framework for Medical Diagnosis of Brain Tumor with an Effective Hybrid Transfer Learning Model. *Diagnostics* **2022**, *12*, 2541. [CrossRef]
30. Anjum, S.; Hussain, L.; Ali, M.; Alkinani, M.H.; Aziz, W.; Gheller, S.; Abbasi, A.A.; Marchal, A.R.; Suresh, H.; Duong, T.Q. Detecting Brain Tumors Using Deep Learning Convolutional Neural Network with Transfer Learning Approach. *Int. J. Imaging Syst. Technol.* **2022**, *32*, 307–323.
31. Khan, A.H.; Abbas, S.; Khan, M.A.; Farooq, U.; Khan, W.A.; Siddiqui, S.Y.; Ahmad, A. Intelligent Model for Brain Tumor Identification Using Deep Learning. *Appl. Comput. Intell. Soft Comput.* **2022**, *2022*, 8104054.
32. Qureshi, S.A.; Raza, S.E.A.; Hussain, L.; Malibari, A.A.; Nour, M.K.; Rehman, A.u.; Al-Wesabi, F.N.; Hilal, A.M. Intelligent Ultra-Light Deep Learning Model for Multi-Class Brain Tumor Detection. *Appl. Sci.* **2022**, *12*, 3715. [CrossRef]
33. Maqsood, S.; Damaševičius, R.; Maskeliūnas, R. Multi-Modal Brain Tumor Detection Using Deep Neural Network and Multiclass SVM. *Medicina* **2022**, *58*, 1090.
34. Khawaldeh, S.; Pervaiz, U.; Rafiq, A.; Alkhawaldeh, R.S. Noninvasive Grading of Glioma Tumor Using Magnetic Resonance Imaging with Convolutional Neural Networks. *Appl. Sci.* **2018**, *8*, 27. [CrossRef]
35. Anaraki, A.K.; Ayati, M.; Kazemi, F. Magnetic Resonance Imaging-Based Brain Tumor Grades Classification and Grading via Convolutional Neural Networks and Genetic Algorithms. *Biocybern. Biomed. Eng.* **2019**, *39*, 63–74.
36. Hemanth, D.J.; Anitha, J.; Naaji, A.; Geman, O.; Popescu, D.E. A Modified Deep Convolutional Neural Network for Abnormal Brain Image Classification. *IEEE Access* **2018**, *7*, 4275–4283. [CrossRef]
37. Ismael, S.A.A.; Mohammed, A.; Hefny, H. An Enhanced Deep Learning Approach for Brain Cancer MRI Images Classification Using Residual Networks. *Artif. Intell. Med.* **2020**, *102*, 101779. [CrossRef]
38. Sajjad, M.; Khan, S.; Muhammad, K.; Wu, W.; Ullah, A.; Baik, S.W. Multi-Grade Brain Tumor Classification Using Deep CNN with Extensive Data Augmentation. *J. Comput. Sci.* **2019**, *30*, 174–182. [CrossRef]
39. Salçin, K. Detection and Classification of Brain Tumours from MRI Images Using Faster R-CNN. *Teh. Glas.* **2019**, *13*, 337–342.
40. Menaouer, B.; El-Houda, K.N.; Zoulikha, D.; Mohammed, S.; Matta, N. Detection and Classification of Brain Tumors from MRI Images Using a Deep Convolutional Neural Network Approach. *Int. J. Softw. Innov. (IJSI)* **2022**, *10*, 1–25.
41. Fangohr, H. A Comparison of C, MATLAB, and Python as Teaching Languages in Engineering. In Proceedings of the 4th International Conference on Computational Science, Kraków, Poland, 6–9 June 2004; Springer: Berlin/Heidelberg, Germany, 2004; pp. 1210–1217.

42. Aruoba, S.B.; Fernández-Villaverde, J. A Comparison of Programming Languages in Macroeconomics. *J. Econ. Dyn. Control* **2015**, *58*, 265–273. [CrossRef]
43. Coleman, C.; Lyon, S.; Maliar, L.; Maliar, S. Matlab, Python, Julia: What to Choose in Economics? *Comput. Econ.* **2021**, *58*, 1263–1288. [CrossRef]
44. Albanese, D.; Filosi, M.; Visintainer, R.; Riccadonna, S.; Jurman, G.; Furlanello, C. Minerva and Minepy: A C Engine for the MINE Suite and Its R, Python and MATLAB Wrappers. *Bioinformatics* **2013**, *29*, 407–408. [CrossRef]
45. Weiss, A.; Elsherbeni, A. Computational Performance of MATLAB and Python for Electromagnetic Applications. In Proceedings of the 2020 International Applied Computational Electromagnetics Society Symposium (ACES), Monterey, CA, USA, 27–31 July 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 1–2.
46. Uddin, M.; Kumar, V.; Yadav, V.K. Comparative Study of the VINCI Robot's Arm End Effector Matrix Using Python and MATLAB. *Mater. Today Proc.* **2021**, *47*, 3761–3764.
47. Çiçek, Ö.; Abdulkadir, A.; Lienkamp, S.S.; Brox, T.; Ronneberger, O. 3D U-Net: Learning Dense Volumetric Segmentation from Sparse Annotation. In Proceedings of the Medical Image Computing and Computer-Assisted Intervention—MICCAI 2016, Athens, Greece, 17–21 October 2016; Ourselin, S., Joskowicz, L., Sabuncu, M.R., Unal, G., Wells, W., Eds.; Springer International Publishing: Cham, Switzerland, 2016; pp. 424–432.
48. Milletari, F.; Navab, N.; Ahmadi, S.-A. V-Net: Fully Convolutional Neural Networks for Volumetric Medical Image Segmentation. In Proceedings of the 2016 Fourth International Conference on 3D Vision (3DV), Stanford, CA, USA, 25–28 October 2016; pp. 565–571.
49. Antonelli, M.; Reinke, A.; Bakas, S.; Farahani, K.; Kopp-Schneider, A.; Landman, B.A.; Litjens, G.; Menze, B.; Ronneberger, O.; Summers, R.M.; et al. The Medical Segmentation Decathlon. *Nat. Commun.* **2022**, *13*, 4128. [CrossRef]
50. Moler, C.; Little, J. A History of MATLAB. *Proc. ACM Program. Lang.* **2020**, *4*, 1–67. [CrossRef]
51. van Rossum, G.; Drake, F.L., Jr. *Python Tutorial*; Centrum voor Wiskunde en Informatic: Amsterdam, The Netherlands, 1995.
52. Ghaffari, M.; Sowmya, A.; Oliver, R. Automated Brain Tumor Segmentation Using Multimodal Brain Scans: A Survey Based on Models Submitted to the BraTS 2012–2018 Challenges. *IEEE Rev. Biomed. Eng.* **2020**, *13*, 156–168. [CrossRef]
53. Summers, D. Harvard Whole Brain Atlas: Www.Med.Harvard.Edu/AANLIB/Home.Html. *J. Neurol. Neurosurg. Psychiatry* **2003**, *74*, 288. [CrossRef]
54. Armato, S.; Beichel, R.; Bidaut, L.; Clarke, L.; Croft, B.; Fenimore, C.; Gavrielides, M. RIDER (Reference Database to Evaluate Response) Committee Combined Report. 2008. Available online: https://wiki.cancerimagingarchive.net/display/Public/Collections% (accessed on 6 February 2023).
55. Kistler, M.; Bonaretti, S.; Pfahrer, M.; Niklaus, R.; Büchler, P. The Virtual Skeleton Database: An Open Access Repository for Biomedical Research and Collaboration. *J. Med. Internet Res.* **2013**, *15*, e2930. [CrossRef]
56. Bakas, S.; Akbari, H.; Sotiras, A.; Bilello, M.; Rozycki, M.; Kirby, J.S.; Freymann, J.B.; Farahani, K.; Davatzikos, C. Advancing the Cancer Genome Atlas Glioma MRI Collections with Expert Segmentation Labels and Radiomic Features. *Sci. Data* **2017**, *4*, 170117. [CrossRef]
57. Simpson, A.L.; Antonelli, M.; Bakas, S.; Bilello, M.; Farahani, K.; Van Ginneken, B.; Kopp-Schneider, A.; Landman, B.A.; Litjens, G.; Menze, B. A Large Annotated Medical Image Dataset for the Development and Evaluation of Segmentation Algorithms. *arXiv* **2019**, arXiv:1902.09063.
58. Menze, B.H.; Jakab, A.; Bauer, S.; Kalpathy-Cramer, J.; Farahani, K.; Kirby, J.; Burren, Y.; Porz, N.; Slotboom, J.; Wiest, R.; et al. The Multimodal Brain Tumor Image Segmentation Benchmark (BRATS). *IEEE Trans. Med. Imaging* **2015**, *34*, 1993–2024. [CrossRef]
59. Berral, J.L.; Aranda, O.; Dominguez, J.L.; Torres, J. Distributing Deep Learning Hyperparameter Tuning for 3D Medical Image Segmentation. In Proceedings of the 2022 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW), Lyon, France, 30 May 2022–3 June 2022; pp. 1045–1052.
60. Zhuge, Y.; Ning, H.; Mathen, P.; Cheng, J.Y.; Krauze, A.V.; Camphausen, K.; Miller, R.W. Automated Glioma Grading on Conventional MRI Images Using Deep Convolutional Neural Networks. *Med. Phys.* **2020**, *47*, 3044–3053. [CrossRef]
61. Korfiatis, P.; Kline, T.L.; Lachance, D.H.; Parney, I.F.; Buckner, J.C.; Erickson, B.J. Residual Deep Convolutional Neural Network Predicts MGMT Methylation Status. *J. Digit. Imaging* **2017**, *30*, 622–628. [CrossRef]
62. Lu, Z.; Bai, Y.; Chen, Y.; Su, C.; Lu, S.; Zhan, T.; Hong, X.; Wang, S. The Classification of Gliomas Based on a Pyramid Dilated Convolution Resnet Model. *Pattern Recognit. Lett.* **2020**, *133*, 173–179. [CrossRef]
63. Yang, Y.; Yan, L.-F.; Zhang, X.; Han, Y.; Nan, H.-Y.; Hu, Y.-C.; Hu, B.; Yan, S.-L.; Zhang, J.; Cheng, D.-L.; et al. Glioma Grading on Conventional MR Images: A Deep Learning Study With Transfer Learning. *Front. Neurosci.* **2018**, *12*, 804.
64. Dong, H.; Yang, G.; Liu, F.; Mo, Y.; Guo, Y. Automatic Brain Tumor Detection and Segmentation Using U-Net Based Fully Convolutional Networks. In *Medical Image Understanding and Analysis*; Valdés Hernández, M., González-Castro, V., Eds.; Springer International Publishing: Cham, Switzerland, 2017; pp. 506–517.
65. Liu, P.; Dou, Q.; Wang, Q.; Heng, P.-A. An Encoder-Decoder Neural Network With 3D Squeeze-and-Excitation and Deep Supervision for Brain Tumor Segmentation. *IEEE Access* **2020**, *8*, 34029–34037. [CrossRef]
66. Rehman, A.; Naz, S.; Razzak, M.I.; Akram, F.; Imran, M. A Deep Learning-Based Framework for Automatic Brain Tumors Classification Using Transfer Learning. *Circuits Syst. Signal Process.* **2020**, *39*, 757–775. [CrossRef]

67. Ronneberger, O.; Fischer, P.; Brox, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. In Proceedings of the Medical Image Computing and Computer-Assisted Intervention—MICCAI 2015, Munich, Germany, 5–9 October 2015; Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F., Eds.; Springer International Publishing: Cham, Switzerland, 2015; pp. 234–241.
68. Hu, Y.; Xia, Y. 3D Deep Neural Network-Based Brain Tumor Segmentation Using Multimodality Magnetic Resonance Sequences. In Proceedings of the International MICCAI Brainlesion Workshop, Quebec City, QC, Canada, 14 September 2017; Springer: Berlin/Heidelberg, Germany, 2017; pp. 423–434.
69. Marcinkiewicz, M.; Nalepa, J.; Lorenzo, P.R.; Dudzik, W.; Mrukwa, G. Segmenting Brain Tumors from MRI Using Cascaded Multi-Modal U-Nets. In Proceedings of the International MICCAI Brainlesion Workshop, Granada, Spain, , 16 September 2018; Springer: Berlin/Heidelberg, Germany, 2018; pp. 13–24.
70. Naser, M.A.; Deen, M.J. Brain Tumor Segmentation and Grading of Lower-Grade Glioma Using Deep Learning in MRI Images. *Comput. Biol. Med.* **2020**, *121*, 103758.
71. Bagyaraj, S.; Tamilselvi, R.; Gani, P.B.M.; Sabarinathan, D. Brain Tumour Cell Segmentation and Detection Using Deep Learning Networks. *IET Image Process.* **2021**, *15*, 2363–2371.
72. Kot, E.; Krawczyk, Z.; Siwek, K.; Królicki, L.; Czwarnowski, P. Deep Learning-Based Framework for Tumour Detection and Semantic Segmentation. *Bull. Pol. Acad. Sciences. Tech. Sci.* **2021**, *69*, e136750.
73. Saeed, M.U.; Ali, G.; Bin, W.; Almotiri, S.H.; AlGhamdi, M.A.; Nagra, A.A.; Masood, K.; Amin, R. ul RMU-Net: A Novel Residual Mobile U-Net Model for Brain Tumor Segmentation from MR Images. *Electronics* **2021**, *10*, 1962. [CrossRef]
74. Sohail, N.; Anwar, S.M.; Majeed, F.; Sanin, C.; Szczerbicki, E. Smart Approach for Glioma Segmentation in Magnetic Resonance Imaging Using Modified Convolutional Network Architecture (U-NET). *Cybern. Syst.* **2021**, *52*, 445–460. [CrossRef]
75. Ghosh, S.; Chaki, A.; Santosh, K. Improved U-Net Architecture with VGG-16 for Brain Tumor Segmentation. *Phys. Eng. Sci. Med.* **2021**, *44*, 703–712. [CrossRef]
76. Kihira, S.; Mei, X.; Mahmoudi, K.; Liu, Z.; Dogra, S.; Belani, P.; Tsankova, N.; Hormigo, A.; Fayad, Z.A.; Doshi, A. U-Net Based Segmentation and Characterization of Gliomas. *Cancers* **2022**, *14*, 4457. [CrossRef]
77. Raza, R.; Bajwa, U.I.; Mehmood, Y.; Anwar, M.W.; Jamal, M.H. DResU-Net: 3D Deep Residual U-Net Based Brain Tumor Segmentation from Multimodal MRI. *Biomed. Signal Process. Control* **2023**, *79*, 103861.
78. Allah, A.M.G.; Sarhan, A.M.; Elshennawy, N.M. Edge U-Net: Brain Tumor Segmentation Using MRI Based on Deep U-Net Model with Boundary Information. *Expert Syst. Appl.* **2023**, *213*, 118833. [CrossRef]
79. Lee, K.; Zung, J.; Li, P.; Jain, V.; Seung, H.S. Superhuman Accuracy on the SNEMI3D Connectomics Challenge. *arXiv* **2017**, arXiv:1706.00120.
80. Zhou, Z.; Siddiquee, M.M.R.; Tajbakhsh, N.; Liang, J. UNet++: A Nested U-Net Architecture for Medical Image Segmentation. In *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*; Stoyanov, D., Taylor, Z., Carneiro, G., Syeda-Mahmood, T., Martel, A., Maier-Hein, L., Tavares, J.M.R.S., Bradley, A., Papa, J.P., Belagiannis, V., Nascimento, J.C., Lu, Z., Conjeti, S., Moradi, M., Greenspan, H., Madabhushi, A., Eds.; Springer International Publishing: Cham, Switzerland, 2018; pp. 3–11.
81. Nodirov, J.; Abdusalomov, A.B.; Whangbo, T.K. Attention 3D U-Net with Multiple Skip Connections for Segmentation of Brain Tumor Images. *Sensors* **2022**, *22*, 6501. [CrossRef]
82. Chen, W.; Liu, B.; Peng, S.; Sun, J.; Qiao, X. S3D-UNet: Separable 3D U-Net for Brain Tumor Segmentation. In *Brainlesion: Glioma, Multiple Sclerosis, Stroke and Traumatic Brain Injuries*; Crimi, A., Bakas, S., Kuijf, H., Keyvan, F., Reyes, M., van Walsum, T., Eds.; Springer International Publishing: Cham, Switzerland, 2019; pp. 358–368.
83. Casamitjana, A.; Catà, M.; Sánchez, I.; Combalia, M.; Vilaplana, V. Cascaded V-Net Using ROI Masks for Brain Tumor Segmentation. In Proceedings of the International MICCAI Brainlesion Workshop, Quebec City, QC, Canada, 14 September 2017; Springer: Berlin/Heidelberg, Germany, 2017; pp. 381–391.
84. Hua, R.; Huo, Q.; Gao, Y.; Sui, H.; Zhang, B.; Sun, Y.; Mo, Z.; Shi, F. Segmenting Brain Tumor Using Cascaded V-Nets in Multimodal MR Images. *Front. Comput. Neurosci.* **2020**, *14*, 9.
85. Giri, C.; Sharma, J.; Goodwin, M. Brain Tumour Segmentation on 3D MRI Using Attention V-Net. In *Engineering Applications of Neural Networks*; Iliadis, L., Jayne, C., Tefas, A., Pimenidis, E., Eds.; Springer International Publishing: Cham, Switzerland, 2022; pp. 336–348.
86. Guan, X.; Yang, G.; Ye, J.; Yang, W.; Xu, X.; Jiang, W.; Lai, X. 3D AGSE-VNet: An Automatic Brain Tumor MRI Data Segmentation Framework. *BMC Med. Imaging* **2022**, *22*, 6. [CrossRef]
87. Siar, M.; Teshnehlab, M. Brain Tumor Detection Using Deep Neural Network and Machine Learning Algorithm. In Proceedings of the 2019 9th International Conference on Computer and Knowledge Engineering (ICCKE), Mashhad, Iran, 24–25 October 2019; pp. 363–368.
88. Han, C.; Rundo, L.; Araki, R.; Furukawa, Y.; Mauri, G.; Nakayama, H.; Hayashi, H. Infinite Brain MR Images: PGGAN-Based Data Augmentation for Tumor Detection. In *Neural Approaches to Dynamics of Signal Exchanges*; Springer: Berlin/Heidelberg, Germany, 2020; pp. 291–303.