An Efficient Deep Learning-based Approach for Glioblastoma Detection from MRI Images

by

Ismail Hossain Saihan
20301159
Umme Mahbuba Tamanna
17301024
Ms Rodsy Tahmid
24341126
Md. Rahadul Islam Fardin
20101363
Fahim Ahamed Romit
20301393

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfilment of the requirements for the degree of
B.Sc. in Computer Science

Department of Computer Science and Engineering
Brac University
October 2024

Declaration

It is hereby declared that

1. The thesis submitted is our own original work while completing degree at Brac University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. We have acknowledged all main sources of help.

Student's Full Name & Signature:

Ismail Hossain Saihan
20301159

Umme Mahbuba Tamanna
17301024

Ms Rodsy Tahmid
24341126

Md. Rahadul Islam Fardin
20101363

Fahim Ahamed Romit
20301393

Approval

The thesis titled "An Efficient Deep Learning-based Approach for Glioblastoma Detection from MRI Images" submitted by

1. Ismail Hossain Saihan (20301159)
2. Umme Mahbuba Tamanna (17301024)
3. Ms Rodsy Tahmid (24341126)
4. Md. Rahadul Islam Fardin (20101363)
5. Fahim Ahmed Romit (20301393)

Of Summer, 2024 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science on October 17, 2024.

Examining Committee:

Supervisor:
(Member)

Mr. Md. Tanzim Reza
Senior Lecturer
Department of Computer Science and Engineering
BRAC University

Co-Supervisor & Program Coordinator:
(Member)

Dr. Md. Golam Rabiul Alam
Professor
Department of Computer Science and Engineering
BRAC University

Head of Department:
(Chair)

Dr. Sadia Hamid Kazi
Associate Professor; Chairperson
Department of Computer Science and Engineering
BRAC University

Ethics Statement

We completed this thesis paper, and it is free of plagiarism.

Abstract

In essence, an abnormal increase of brain cells is referred to as a brain tumor. Tumors come in two varieties: benign (non-cancerous) and malignant (cancerous). Cancerous tumors can originate in the brain itself (Primary) or spread from elsewhere from other parts of the body as well (secondary or metastatic tumors). One of the most aggressive and malignant forms of brain tumor is Glioblastoma which is also known as glioblastoma multiforme (GBM). Glial cells are the supportive cells in the brain which is the main origin of GBM tumors. The rapid growth of GBM and its tendency to spread into nearby brain tissue makes complete surgical removal of the tumor challenging. The main purpose of this study was to build an efficient model with the application of deep learning techniques to detect glioblastoma accurately. We implemented a binaraized version of ResNet18, VGG16, and DenseNet121 with the Binary Weight Networks (BWN) approach. This conversion to binary values saved almost 30x memory. In Binary Weight Networks (BWN) the weights are binary value but the inputs are not binarized, input data remains full-precision format. Applying binary operation in convolution layers helped 40x faster operations in convolution compared to full precision operations. This memory savings will help us to run those models in real-time only using CPU rather than heavy GPU. Our simple binary models are efficient and accurate also on detecting Glioma. We evaluated our model's performance with TCGA-GBM and IXI dataset using accuracy, confusion matrix, and ROC curve matrics. For ResNet18 we got 89% accuracy. For DenseNet121, we got 85% accuracy. And for VGG16, we got 87% accuracy. The classification with a Binary Weight Network version of ResNet18, DenseNet121, and VGG16 is as closely accurate as the full precision. We compared our binary models with full-precision models, which gave us a balance between accuracy and efficiency, with a 33x reduction in model size and 30x memory saving.

Keywords: Glioblastoma detection, MRI, Binarized Neural Networks, ResNet18, VGG16, DenseNet121, Binary-Weight-Networks, Deep Learning.

Dedication

This study is devoted to the tenacious nature of the Bangladeshi people, whose health and welfare motivate us to come up with novel ideas and gradually close the gaps in healthcare. To the innumerable people who kindly offered their time, expertise, and experiences; as a result, this research is a reference to your priceless initiatives.

Acknowledgement

Firstly, all praise to the Great Allah for whom our thesis have been completed without any major interruption.
Secondly, to our respected supervisor Md. Tanzim Reza sir for his kind support and advice in our work. He helped us whenever we needed help.
Thirdly, we appreciate the guidance and feedback provided by our honorable co-supervisor Professor Dr. Md. Golam Rabiul Alam sir.
And finally to our parents without their throughout sup-port it may not be possible. With their kind support and prayer we are now on the verge of our graduation.

Table of Contents

List of Figures

List of Tables

Nomenclature

The next list describes several symbols and abbreviations that will be later used within the body of the document.

AUC	Area Under the Curve
BWN	Binary-Weight-Networks
BNNs	Binarized Neural Networks
IXI	Brain Tumor Segmentation
TCGA-GBM	Cancer Genome Atlas Glioblastoma Multiforme
CPUs	Central Processing Units
CMOS	Complementary Metal-Oxide-Semiconductor
CNN	Convolutional Neural Network
DNN	Deep Neural Network
FPGAs	Field-Programmable Gate Arrays
FPS	Frames Per Second
HQC-CNN	Hierarchical Quantized Convolutional Neural Networks
MLP	Multilayer Perceptron
LWB	Layer-Wise Binarization
LSTM	Long Short-Term Memory
MNIST	Modified National Institute of Standards and Technology
QCNN	Quality-Controlled Convolutional Neural Network
RPReLU	ReAct Parametric Rectified Linear Unit
RSign	ReAct Sign
RRAM	Random-Access Memory
ROC	Receiver Operating Characteristic
SVM	Support Vector Machine
VGG16	Visual Geometry Group 16-layer
WHO	World Health Organization
CIFAR-10	Canadian Institute for Advanced Research dataset 10 classes
SVHN	Street View House Numbers

Chapter 1

Introduction

Gliomas are a wide range of brain tumors originating from glial cells. Tumors are graded based on the origin of the cells and malignancies. The WHO classified glioma in grading from I to IV, where grade IV glioma is the most malignant [5] . Glioblastoma multiforme is the most frequent and highly aggressive malignant primary neoplasm in adult brain tissue, being categorized into Grade IV glioma. It is characterized by rapid growth, invasive potentiality, and resistance against any sort of therapeutic approach, which results in poor outcomes for patients [5] . The annual incidence rate of glioblastoma is about 3.2 per 100,000 in the United States. Despite aggressive practice, the prognosis of glioblastoma remains disheartening, with an average survival of 12 to 15 months following diagnosis [31]. Early detection of GBM is of paramount importance in improving therapeutic outcomes; however, heterogeneity and invasion by GBM render this very challenging [35]. Because of the fast-moving nature of GBM, early diagnosis and intervention has become key. Advanced imaging technologies, therefore, have a very vital role in the identification of tumor characteristics and conducting treatment.

MRI Technique and its Clinical benefit in Brain Imaging is an imaging technique in a non-invasive way and one of the most commonly used techniques for brain tumor diagnoses. MRI gives better soft-tissue contrast compared to other imaging techniques like CT scans and hence is very useful for the detection and evaluation of brain tumors [33]. Several MRI sequences are employed for different diagnostic purposes in glioma detection: T1-weighted imaging gives information about the detailed anatomical structure. Edema and tumor infiltration can be identified with sequences such as T2-weighted and FLAIR. DWI measures the diffusion of water molecules, hence it helps in calculating tumor density. PWI estimates blood flow inside the tumor and thus aids in grading gliomas [11]. This process is time-consuming, and interobserver variability also exists in the delineation performed by radiologists. The complex infiltrative pattern of GBM growth further complicates the correct delineation of the tumor boundary and the estimation of the actual extension of the disease [29]. The manual analysis of MRI Images is very time-consuming and, besides that, prone to interobserver variability between radiologists. Infiltrative growth is one of the hallmarks of GBM; hence, the delineation of tumor boundaries and assessment of the whole extent of disease is further complicated [44]. Given the difficulties in manual observation, various machine learning and AI-based automated detection techniques have attracted more interest. These have the added

advantage of increased precision and speed, thereby assisting clinical management in formulating appropriate treatment strategies [37].

The majority of the research dealing with glioblastoma employs traditional segmentation and radiomics approaches that rely heavily on manual interpretation of MRI images. Most of these techniques demand high expertise and are therefore very time-consuming, hence leading to variability in diagnosis [38]. Deep learning applied to medical imaging is revolutionary in that it enables the automation of MRI scan analysis. Convolutional neural networks (CNNs) have emerged as a strong tool in this study therefore seeks ways of improving the efficiency and accuracy of the conventional methods of detection of glioma [19]. However, several challenges still persist, such as the need for huge amounts of annotated data and the failure of models to generalize across various patient populations.

Despite improvements in the field of glioblastoma detection, current approaches still have several limitations. Most of the models presented suffer from problems of efficiency and scalability. Most of the presented models have been computationally expensive to train, and such computational resources are not easily available for the routine use of clinical practice. This is indicative of the need to develop a more efficient, more accurate, and highly scalable approach to glioma detection. Binarized neural networks (BNNs) may represent a promising alternative; they can reduce model complexity with minimal loss of accuracy [42]. The presented work aims to develop a lightweight model, suitable for use on a mobile device, by exploring the peculiarities of BNNs, in order to enhance accessibility and usability for healthcare professionals.

In contrast to CNN, a binary neural network uses weights and activation functions where the values are between -1 and +1. CNNs use a 32-bit floating point operation for the parameters where BNNs operate by transforming the weights and activations into binary values through bitwise operations such as XNOR and popcount [20]. This binarization process simplifies the complex mathematical operations of CNNs, and thus, it results in faster computational speed with lower memory requirements [20]. Tasks like pattern recognition, image classification, object detection demand a significant number of resources which are quite impossible for resource-constrained devices to provide. In such scenarios, BNNs offer an alternative to reducing high-precision calculations, an easy deployment on embedded systems or IoT platforms [45], [4]. This trade-off, however, causes a reduction in accuracy compared to CNNs. Yet, in this modern era – through real-time analysis of medical imaging (e.g., MRI scans) for disease detections like brain tumors (Glioblastoma), edge computing in clinical applications, BNNs outweigh the loss in precision through efficiency, memory usage of CNNs without substantially compromising accuracy [4].

1.1 Problem Statement

Glioblastoma is an especially worrisome and deadly form of brain cancer for two reasons: it proceeds rapidly, is hard to detect early on, and has a highly malignant structure. That is, while glioblastoma accounts for only about 15% of all brain tumors [24]. it has an inordinate morbidity and mortality impact as evidenced by

the WHO median survival time of 15 months from diagnosis. Current diagnostic methods-usually consisting of an MRI scan followed by manual analysis-are time-consuming and prone to variability [43], sensitive to the experience and precision of the medical expert who analyzes its results. With machine learning and deep learning models that automatically handle portions of the diagnostic process becoming widespread, computational cost still remains a major concern. Large-size full precision deep learning models have to utilize high-performance GPUs with high memory size and energy consumption, similar to conventional convolutional neural networks (CNN). Such models are actually difficult to deploy in such low resource environments [24], which is precisely where we need these diagnostic tools to be the most efficient and automated. This leaves a big gap in access to timely and accurate glioblastoma diagnosis, which is especially a problem when the healthcare facility served in rural clinics, smaller hospitals, or places in developing nations might not have the infrastructure to actually run these resource-intensive models. Improved patient outcomes depend on early detection, as research states that if intervention occurs early, survival rate can be higher by up to 30% . However, access to advanced diagnostic tools is currently limited in many parts of the world.

1.2 Research Objective

To address this problem, we explored Binary Neural Networks (BNNs), ultralight neural networks that perform almost equally well as traditional models but with orders of magnitude less computational and memory demands. BNNs replace the standard full-precision floating point weights and activations with binary representations, thereby reducing model storage size by an order of magnitude and simplifying the calculation complexity of the model. This size reduction further allows BNNs to fit in low-resource medical settings by allowing activation on a device with limited hardware, such as mobile phones or embedded systems. Our BNN is based on renowned architectures like ResNet18, DenseNet121, and VGG16 as a backbone to maintain the accuracy and performance of the model. Skip connections used by ResNet18 allow deeper networks to be used without some of the customarily associated problems of over-deep models like vanishing gradients. Since ResNet18 is a robust architecture for conversion into a BNN framework, it makes sense to take the balance of these trade-offs between the numerical computational efficiency and the model accuracy. Since ResNet18 can be easily adapted to a binary framework, our model can effectively detect glioma using far less computational power than before making it possible to deploy the model itself in resource-constrained environments and still get good diagnostic accuracy. Our overarching goal is to take advanced medical diagnostics off the high-end hardware grid, particularly in places where that equipment is unavailable, to increase early detection rates and save lives. Our system, a model combining BNNs with ResNet18, DenseNet121, and VGG16, provides a unique solution to the existing problem of accessibility and efficiency in glioma detection [24] to enable low-cost, scalable, and efficient diagnostic tools.

Chapter 2

Literature Review

Author Rastegari et al. [6] introduces two binary convolutional neural networks: Binary-Weight-Networks (BWN) and XNOR-Networks, for image classification purpose. BWN binaries only the weights of convolutional layers during forward pass, reducing memory usage by 32 times and simplifying operations to addition and subtraction. XNOR-Networks extend this by binarizing both weights and inputs of convolution and fully connected layers Layers that are completely related , enabling convolutions to be approximated using XNOR and bitcounting, leading to a 58 times speedup. Both models are trained from scratch without pre-trained networks, using binarized values during forward and backward passes while keeping real-valued parameters for updates. A scaling factor is computed for each layer to preserve accuracy. The models were tested on the ImageNet dataset. The Binary-Weight-Network version of AlexNet achieved 56.8% top-1 and 79.4% top-5 accuracy, nearly matching the full-precision version. XNOR-Net, with both inputs and weights binarized, achieved 44.2% Top-1 and 69.2% Top-5 accuracy. Despite the drop of accuracy, the efficiency gains are significant. XNOR-Net outperformed previous binarization methods like BinaryNet, which achieved only 27.9% Top-1 accuracy. Applied to deeper models like ResNet18, the binary-weight version reached 60.8% top-1 and 83.0% Top-5 accuracy, in comparison with the 69.3% and 89.2% for full-precision ResNet18.These findings outcome describe XNOR-Net's potential for efficient real-time applications on devices which have limited resources.

Author Bulat et al. [17] discusses the limitations of binary neural networks in accurately preserving the predictive power of full-precision models, particularly when applied to complex datasets such as ImageNet, and introduces methods to enhance both the representation and performance of such networks. Addressing this issue, the authors introduced an enhanced training process for binary neural networks, in which activations and weights are binary, targeting the improvement of scaling factors used in XNOR-Net. Instead of analytically calculating separat factors of e scaling for activations and weights, XNOR-Net++ combines these factors into a single scaling factor that is learned through backpropagation. This modification not only improves the accuracy but also optimizes computational efficiency by facilitating the capture of statistical representation of data and better test time analytical calculations.The paper explores several ways to construct the shape of the scaling factor, improving the network's ability to approximate the output of binary convolutions without increasing computational complexity during inference.

XNOR-Net++ is evaluated using deep residual networks, specifically ResNet18, and shallow networks like AlexNet on the challenging ImageNet dataset. For ResNet18, XNOR-Net++ achieves a Top-1 accuracy of 57.1% and a Top-5 accuracy of 79.9% representing a significant improvement of over 6% compared to the original XNOR-Net. For AlexNet, it achieves 46.9% Top-1 and 71.0% Top-5 accuracy, showing more modest improvements. These results accordingly show that this approach closes the accuracy gap while preserving binary neural network efficiency improvements.

Author Helwegen et al. [12] highlights the challenge of training binary neural networks from scratch, emphasizing that existing methods often rely on full-precision pre-training. They propose a competitive approach that trains binary networks without this dependency, improving performance on complex datasets like CIFAR-10 The authors proposed two modified versions of ResNet and DenseNet. For ResNetE, firstly, the number of connections were increased by reducing the block size leading to twice the amount of shortcuts keeping the layer amount the same. For DenseNetE, they modified its architecture that doubles the number of layers while halving growth rates to enhance information flow. Binary convolution layers are implemented using XNOR and popcount operations, reducing computational costs and memory usage. No weight decay is used, and scaling factors are evaluated but found unnecessary. Tested on MNIST, CIFAR-10, and ImageNet, ResNetE-18 achieved 87.6% accuracy on CIFAR-10 and 56.7% on ImageNet, while DenseNetE-21 achieved 90.3% and 57.1% respectively. These results outperform previous binary models like XNOR-Net and BiReal-Net, demonstrating that BNNs can achieve state-of-the-art accuracy without complex fine-tuning or pre-trained models, making them highly efficient for resource-limited devices.

Author Tang et al. [10] points out the issues of training a binary neural network with high compression rates and high accuracy on large-scale datasets like imageNet. Even though BNNs reduce memory and computational costs by binarizing weights and activations, it suffers from low accuracy and limited compression rates when a large dataset is fed into it. The authors identified some key points that contradicts the nature of BNNs, such as – instability caused by frequent weight change due to high learning rates, the failure of ReLU activation in binary layers and inappropriate regularization that drives weight to zero. To resolve these issues, they suggested solutions like – lowering the learning rate to stabilize training, replacing ReLU with PReLU to better absorb the scale factor into the activation function, using a custom regularization term to encourage bipolar weights instead of zeroing them out. In addition to it, the authors highlighted the difficulty of binarizing the last layer that hinders compression rates. And as a solution to this problem, they proposed a scaling layer to manage the vast range of outputs from the final binary layer. Moreover, they applied the idea of multiple binarization to improve activation approximation without causing reduction in efficiency. All these strategies demonstrated a significant improvement in accuracy as well as compression rate in ImageNet dataset.

In this paper, author Prabhu et el. [15] investigates the challenges of balancing the advantages of network binarization with the significant loss in accuracy it can produce, which is, fully binarizing the network often results in reduction in accuracy. To address this issue, the authors binarized only a selective amount of layers while other

layers remain in full precision. In this hybrid binarization strategy, they developed a metric that mutually considers binarization error and computation cost to decide which layers should be binarized. A partitioning algorithm is introduced using this metric that identifies layers to binarize based on K-Means clustering of layer-wise error metric. This study was conducted on ImageNet dataset like TU-Berlin and sketch dataset -Sketchy, applying hybrid binarization approach to models – AlexNet, ResNet18, and Sketch-A-Net. Hybrid binarization method led to approximately 3-4% improvements on ImageNet dataset and more than 8% gain in accuracy on sketch dataset over fully binarized models. The key insight is that, the last layer of the network which contains many parameters, can be binarized with little accuracy reduction which allows further memory savings.

Wang et el. [16] described the problems they confronted on neural network binarization. Traditional deep learning models, like VGG16, need a substantial amount of resources limiting its deployment on mobile and embedded devices. So this paper introduced a novel binarization method called Layer-Wise Binarization (LWB), to efficiently approximate real-valued parameters using binary codes while minimizing performance loss. The core concept of this work is the layer wise priority in the binarization process, which considers the impact of binarizing layers produced on the model's compression ratio and performance. Instead of binarizing all layers simultaneously, the layer wise binarization method selectively binarized layers in inverse order of their depth. The reason for this is that binarizing all the layers all together can lead to significant performance degradation and lower accuracy rate. Where, this selective layer binarization approach allows layers which have much more impact on the compression ratio with minimal performance loss to be binarized first, using a block coordinate descent scheme to iteratively minimize the discrepancy between real-valued weight and their binary approximations. During each training step, when a specific layer is binarized, its previous layers are kept fixed, which enables a flexible trade-off between performance and compression. Such trade-off allows the model to stop binarizing layers if the performance loss exceeds a certain threshold, resulting in better control over the balance between network size and accuracy. This method was validated through experimentation on pedestrian detection benchmarks (Caltech and INRIA datasets) using the YOLOv2 detection framework. Their LWB method projected faster convergence compared to binary weight networks with a lower miss rate of 26.7% and 13.8% respectively in Caltech and INRIA dataset as well as smaller model size.

Li et el. [18], designed natural scene text interpretation in mobile edge computing environment using a binarized convolutional encoder-decoder network. The challenges of their work were addressing real-time processing, power efficiency, and latency, keeping in mind that traditional GPU-based workings are unsuitable because of their high power consumption and latency in communication with off-chip memory. To overcome this, an ASIC-based accelerator is proposed by the authors, as so to achieve low-latency, energy-efficient, real-time scene text recognition on resource-constrained edge devices and embedded systems, such as mobile phones. The main work of the paper lies in binarizing both the weights and activations of the neural network, transforming computationally expensive floating point operations into bitwise xnor operations, which allowed massive parallelism in the computation,

especially reducing both memory requirements and computational complexity. Besides, the authors used an encoder-decoder architecture, where the encoder extracts high-level features and decoder generates pixel-wise classification maps. To further optimize performance for edge computing, the entire system was designed to store all parameters and intermediary results on-chip, eliminating the need for off-chip memory communication, which can lead to power consumption and latency issues. The architecture was implemented using 40nm CMOS technology, achieving a frame rate of 34 FPS with a latency of 40 ms for interpreting 128x32 text images. The energy efficiency stood out, reaching 698 GOP/s/W in real use cases and peaking at 7825 GOP/s/W, making the system 7x more energy efficient than its GPU-based counterparts. Moreover, the system could classify text regions in natural scene images with over 90% accuracy on well-known datasets like ICDAR-03 and ICDAR-13. Their work highlights the potentials for binarized neural networks to enable efficient and scalable edge computing solutions.

Courbariaux et el. [2], offers a promising study of a more hardware efficient deep learning method, by using binary weight during training propagations. Their main motivation was to resolve the facts of computational limitations when training deep neural networks on large dataset as well as to cater to the needs of deploying these networks on low-power devices. Here, they replaced the real-valued weights with binary weights during propagation while training the DNNs via stochastic gradient descent. The authors presented two techniques for binarizing weight, first is, deterministic binarization, where the sign of the real-valued weight determines the binary value; and second one is stochastic binarization, where a probability function determines the binary value based on the magnitude of the weight. During the training process, real-valued weights are updated using gradients, but forward and backward propagations are carried out with their binary versions, which ensure efficient computation. One important factor is, this paper can be seen as a regularization, as the noise introduced by binary discretization has a similar effect to methods like Dropout, enhancing the model's generalization. The authors validated this paper on three datasets: MNIST, CIFAR-10, and SVHN. Here they employed a multilayer perceptron (MLP) for the permutation-invariant MNIST dataset and a convolutional neural network for CIFAR-10 and SVHN. The stochastic version of the method achieved state-of-the-art performance without requiring floating-point precision during propagation. In this paper, it achieved a test error rate of 1.18% on MNIST, 8.27% on CIFAR-10, and 2.15% on SVHN, nearly blending with the performance of more traditional DNNs while using significantly fewer computational resources.

Author Phan et el. [27] addresses the challenges of designing an efficient Binary Neural Networks for mobile platforms. BNNs are highly compact and computationally efficient, nevertheless, they face performance limitations due to the constraints of weight binarization and activations, which reduces the representation capability of the network. In this paper, the authors identified a key issue: depth-wide convolution in MobileNet, when binarized, results in a narrow output range, which limits the power and can lead to performance degradation of the model. Here, fully binary convolution layers could potentially improve the range of output values but it causes a high computational cost. So the challenge was balancing between com-

putational efficiency and preserving accuracy in binary models for mobile devices. Now, to address these issues, a strategy of evolutionary search mechanism has been introduced in this paper for advancing the binarization process of MobileNet, dividing the method into three steps: pre-training a binary network with random group combinations, evolutionary search for optimal group combinations and retraining the network from scratch. In the first step, binary convolutional neural networks are pre-trained using a wide range of random group convolutions at each layer, which assists the model preserve necessary information via the network. In the second step, an evolutionary algorithm is applied to search for the best group convolution configurations. Group convolution gives a middle ground between depth-wise and fully connected convolution, offering a wider output range without excessively increasing computational cost. This step focuses on optimizing accuracy while maintaining constraints on the number of floating-point operations (FLOPs). Moreover, it stands out as a significant component of the whole method since it allows exploration across various group configurations ensuring the best possible trade-off between accuracy and computational cost. Finally, the third step describes that the network is trained from scratch using discovered group combinations as the optimal group configurations are being identified in the second step. This step ensures that the final binary model is well-optimized in case of accuracy as well as computational efficiency. This method was experimented on a large-scale ImageNet dataset, where the final model obtained 60.09% Top-1 accuracy, outperforming CI-BNN at the same computational cost. Furthermore, their approach significantly reduced the model's complexity and computational requirements making it more suitable for deployment on mobile devices.

Bethge et al. [21] in this paper proposed a novel architecture, MeliusNet, to enhance the efficiency and accuracy of BNNs. The core innovation in MeliusNet is the changed structure of Dense Blocks and Improvement Blocks. Inspired by BinaryDenseNet, Dense Block concatenates newly computed channels to feature map, increasing the feature capacity. The Improvement Block enhances the quality of these newly added features by applying residual connections to improve the new features without affecting the previously computed ones. Alongside with it, the authors proposed the redesign of the initial layers in the network which are not binarized in typical BNN architecture. Rather, they replaced the computationally expensive 7x7 convolution layer with multiple grouped 3x3 convolutions, which reduced the number of operations more than 40% without any drop in accuracy. To further reduce the number operations, they introduced a Grouped Stem architecture that uses grouped convolutions in the initial layers. Moreover, using the Adam optimizer and specific training strategies added improvements in their work. Their experiment was conducted on ImageNet dataset which demonstrated that MeliusNet achieves better performance compared to previous BNN architectures, which specially matched the accuracy of MobileNet-v1. Also, different configurations of MeliusNet e.g., MeliusNet22, MeliusNet42 were displayed to achieve Top-1 accuracy levels comparable to MobileNet-v1 while using significantly fewer operations. On the other hand, Z et al. [25], in this paper, leaned toward the solution with a generalized activation function strategy to address the stated issues. They proposed a baseline network design with a modified MobileNet-v1 structure that uses parameter-free shortcuts bypassing all intermediate convolution layers and allows

binary computations to flow unhindered. The purpose of this shortcut is that it helps the network to propagate more accurate real-valued feature maps by reducing computational overhead. Furthermore, the authors proposed two novel activation functions – ReAct-Sign (RSign) and ReAct-PReLU (RPReLU). Rsign generalized the sign function through a learnable threshold that helps the network differentiate between meaningful features and noise. Likewise, RPReLU extends the PReLU function that enables the network to adjust activation values more precisely. Additionally, a distribution loss and downsampling with channel concatenation were used to further optimize the network. Their proposed ReActNet-A achieved 69.4% Top-1 accuracy on ImageNet dataset, outperforming Real-to-Binary and Melius-Net29 by 4.0% and 3.6% respectively using half the computational resources, and ReActNet-C achieved 71.4% Top-1 accuracy closing the performance gap with real valued networks to 3.0%.

Kung et el. [13] discusses the application of BNNs for detecting embedded objects, whereas a lot of interest has recently been taken in the use of BNNs because they manage to reduce consumptions of memory and computations at high accuracy. The applications of BNNs can be extended to image recognition, medical diagnostics, and their potential use in low-power computer vision applications. Kung et al. proposed a work called "Efficient Object Detection Using Embedded Binarized Neural Networks" on human detection from IR images with a system based on BNNs. In this work, it is demonstrated that BNNs can be used in order to reduce the memory and energy consumption while performing comparably to a conventional 32-bit floating-point model but with much lower computation and overhead. The study highlights a 4x speedup and three orders of magnitude improvement in energy efficiency compared to GPU-based systems, hence underlining the potentiality of BNNs in the embedded platform. Deep learning models like LeNet5, AlexNet, and DeepFace contain large parameters, which easily lead to serious energy efficiency and memory constraints in an embedded system. Contrarily, BNN can alleviate the memory problem by binarizing weights and activations, which leads to huge memory reductions with high accuracies for image classification and object detection tasks. The work here proposes the architecture for further reductions in BNNs targeting low-power devices, considering memory efficiency. It is maintained that FPGAs can afford to support binary neural networks, hence it is a feasible alternative on embedded platforms such as the Zynq-7100. Feasibility has been demonstrated with the use of BNNs in human detection on infrared images, achieving accuracy above 98% without noise and 96% with injected noise. These networks provide reduced memory and energy efficiency and are among the best options for embedded object detection systems, particularly in applications where power is a limitation. Embedding these systems onto smaller FPGA boards for fully embedded applications will be further investigated as an area of primary continued development.

Zhang et el. [41] comprise the vulnerability of the CAN protocol to cyber-attacks. It mainly discusses the design of a new IDS, based on BNN. The IDS aims to enhance security and operational efficiency, and its goal is to deploy it in real-time on in-vehicle networks. The authors propose a BNN for fast detection processes, keeping memory and energy consumption as low as possible with high accuracy in detection. Furthermore, it can be further optimized using FPGAs, hence making it

apt for embedded automotive environments. In-vehicle networks, such as CAN, are at increasingly higher risk regarding cybersecurity due to the higher proportion of connected and autonomous vehicles on the road. Traditional CAN protocols do not have embedded security mechanisms and are thus open to all kinds of attacks, such as message injection, spoofing, and replay attacks, which may pose serious threats against vehicle safety. This work is motivated by the need to secure in-vehicle networks through lightweight, low-latency IDS solutions that would identify this kind of attack in time with minimal resource consumption. Other machine learning-based IDSs, such as those developed with ANN and CNN, although very accurate in their detection, are computationally intensive and, therefore, have very high latency and energy usage. In that respect, the authors are proposing an IDS based on BNN. In the case of BNN, weights and activations are binarized to either of the two values, +1 or -1, which significantly brings down computational and memory requirements with only a slight degradation in accuracy. Executions of the proposed BNN-based IDS will be performed on CPUs, GPUs, and FPGAs, respectively. FPGA acceleration is useful in applications that require real-time processing due to low power consumption and parallel handling of several tasks. The IDS does not require any DBC files describing the specific CAN messages provided by the vehicle manufacturer, and this helps to support a great variety of vehicle models and network settings. The experimental results show that the BNN-based IDS outperforms the conventional neural network-based IDS in detection time with comparable accuracy. The implementation on FPGA hardware will make it even better, contributing to low latency and power consumption with high performance in its detection. It also highlights that the BNN-based IDS may not detect new or unseen attack types, which calls for further research on adversarial techniques and hybrid models of detection that can adapt better to ever-changing threats. In conclusion, it is obvious that the proposed BNN-based IDS can promise a solution to improve cybersecurity for in-vehicle networks in terms of reduction of detection latency and resource consumptions. Future work will investigate ways for improving the performance of the proposed system against hybrid attacks and explore avenues to include semantic features of CAN traffic in the IDS model.

Vreca et el. [34] researched about Detecting Network Intrusion Using Binarized Neural Networks (BNNs).In the IoT, there is an increase in connected devices or attack surfaces; hence, network security remains challenging. Network Intrusion Detection Systems are considered the basic mechanisms for detecting malicious behavior based on network traffic. Classic NIDSs are computationally very expensive and, therefore, less feasible in resource-constrained environments such as the IoT. BNNs are a hardware-friendly solution that can run on low-end devices like FPGAs. Actually, the BNN provides an advantage in memory consumption and inference speed, making it possible for deployment in hardware platforms. These are efficient in handling network traffic data coming from various IoT devices for real-time packet classification into normal or malicious packets. The biggest drawback with them is the low accuracy compared to their full precision models. It demands further research in improving the performance of BNN while keeping hardware efficiency.

Gerutti et el. [22] shows how sound is detected using BNN on tightly power constrained IoT devices. The SED system is considered a major enabler in smart

cities and consumer electronics, which is mostly reliant on conventional deep learning models. This involves a few challenges such as high memory and processing power with high energy consumption. This work discusses reducing such challenges of DNNs by introducing Binary Neural Networks: providing binary values instead of full-precision numbers, making the computation simpler. Thus, hydraulic platforms like the RISCV-based GAP8 microcontroller were developed that efficiently ran BNNs with HW-popcount-like specialized instructions for fast execution of binary convolutions. The architecture helps in improving energy efficiency and makes BNN apt for SED real-time tasks on edge devices. Despite the manifest power and memory efficiency advantages, BNNs still incur a small loss of accuracy compared to full-precision networks. This is usually within reasonable bounds on average: 7-18% below the full precision baselines. The BNN-based SED systems share a similar structure to the classical DNN model while executing binary convolutions in order to cope with reduced precision of weights and activations. This was optimized for low power execution, allowing the rest of the layers to be binary, placing real-valued layers only where necessary to avoid a loss of accuracy. Experiments reveal that GAP8 deployments of BNN are much more energy-efficient and result in higher throughput, showing how binary neural networks have big potential in ultra-low-power devices for advanced IoT applications.

Shell et al. [9] presents a portable device that uses neural networks to classify malignant and benign tissues based on bioelectrical properties, providing a non-invasive solution for cancer detection. ANNs are meant to analyze complex, nonlinear data patterns, which puts them into a perfect position to detect the minute differences between malignant and benign tissues in cancer diagnosis. The six-feedforward backpropagation neural networks were used by the researchers for classifying tissue samples, each of them trained using impedance data from excised tissue specimens. The accuracy achieved was impressive, with sensitivity and specificity both reported at 100%.The portable device utilizes tetrapolar blackened platinum electrodes to capture 47 impedance data samples, analyzed by BNNs. For the analysis, the Cole-Cole model and Nernst's equation are used to reflect the bioelectrical equilibrium potential of the membrane-a very important parameter in determining malignancy. The advantages provided in clinical environments when there is little time and resources are several-fold: this can provide cancer detection without biopsy or extensive laboratory analysis. These results prove that with neural networks, one can get both high sensitivity and specificity for cancer detection, with accuracy comparable to traditional histopathology. Because it is portable, this device opens up new avenues of use in several clinical settings both at the hospital and clinical level, and in remote clinics.

Scarpace et el. [8] shows a comparison between Binarized Neural Networks (BNNs) and Convolutional Neural Networks (CNNs). Large data amount growth induces more research into methods for predictive accuracy. High performance in image and speech recognition has granted CNN an important place. Binarized Neural Networks are an alternative architecture offering faster computational efficiency by merely using binary weights and activations. This paper presents a performance comparison of BNN and CNN on MNIST and CIFAR-10 datasets, with focus on the dropout technique for regularization. The CNNs have been able to classify im-

ages with high performance using their convolutional layers, which can learn data with spatial hierarchies. However, one major problem for deep neural networks such as CNN is overfitting, where the model simply memorizes the training data instead of generalizing it on new data. Overfitting naturally led to the development of regularization techniques such as dropout, which randomly "drop" neurons during training in a bid to force the network to learn a redundant representation and thereby make it improve its generalization capability. BNNs represent each weight and activation by using a binary value, which reduces complexity and computational cost because it can achieve faster processing speed and less memory. However, their binary nature can easily reduce their expressiveness and make the already difficult task of matching CNN performance on complex tasks. One avenue taken for generalization improvement of BNNs is the integration of dropout, especially in cases with datasets where overfitting is very probable. While BNNs have computational advantages, some areas exhibit trade-offs in performance, especially on the classification problems related to complex datasets like the CIFAR-10. Generally, BNNs take more time to train compared to CNNs because binary weights pose optimization challenges. When dropout and batch normalization are used in tandem, the performance achieved is close to that of conventional CNNs. Optimizing the dropout rates in BNN, further reduction of training time, and using different datasets for widening the applicability range of BNN on diverse tasks will be one of the future directions. Additionally, further studies on filter size will shed light with respect to execution time and further helpful hints toward real-world deployment of BNNs.

Penkovsky et el. [26] comprises In-Memory Resistive RAM Implementation of Binarized Neural Networks for Medical Applications. This has been the era of rapid growth of deep learning, hence bringing about state-of-the-art machine learning applications in several fields such as healthcare. However, such deep neural networks have posed large energy and memory requirements when being deployed on edge devices. This has become a big barrier. To address this challenge, BNNs implemented on RRAM have shown promise. BNNs are a special variety of neural networks, which reduce computational complexity and thereby reduce memory. Hence, they are also ideal candidates for edge devices where energy consumption is a primary concern. They can efficiently process medical signals with ensured privacy at the edge and with low latency, thereby reducing the memory footprint of neural networks. Partial binarization of neural networks presents a balanced approach in optimizing memory usage without significant loss in performance. With mature RRAM technology and better binarization techniques, BNNs will surely revolutionize the deployment of machine learning models for healthcare applications with high demands on low power, privacy sensitivity, and real-time performance.

Peddinti et el. [32] researches about Evolution in Brain Tumor Diagnosis and Detection. Brain tumor detection has grown into an important area of current research due to the potential impact on patient survival rates. The advancement in diagnostic imaging, especially Magnetic Resonance Imaging, has increased the rate of detection of brain tumors. Due to increasing complexities and volumes, there has resulted in a dire need for more sophisticated techniques to tackle such data to overcome various overheads and delays. This literature review is going to determine how techniques have evolved in the quest to improve accuracy, efficiency, and speed

for brain tumor detection, with a bias on integrating machine-learning techniques into the diagnostic system. Various MRI-based detection and segmentation methods have been made in the direction of having better accuracy with least computational burden since the datasets are usually large. Traditional segmentation methods, like algorithms using watershed techniques and region-based methods, suffer from certain limitations when dealing with the complex and often irregular shapes of tumors. Graph-based methods and curvature-based segmentation techniques show some promise, yet further refinement is still needed to reduce false positives and enhance boundary detection. Some of the algorithms in machine learning which have been employed to enable automatic classification of brain tumors based on MRI data include ANN, deep models of learning, and CNN. The methods are also suffering from computational challenges related to processing speed, classification accuracy, and scalability of the system. Future research likely focuses on advanced deep learning models, adaptive filtration techniques, multi-scale segmentation, and advanced feature extraction methods.

Qin et el. [28] researches about Recent Progress on Memristive Convolutional Neural Networks for Edge Intelligence. The fast growth in data and computing technologies means fast development in the field of artificial intelligence, specifically in convolutional neural networks. However, edge computing requires more efficiency and real-time processing, a challenge for traditional cloud-based AI systems. Among all those, memristor is a promising technology because of its nonvolatile nature and to perform in-memory computing. Memristive CNNs enjoy many significant advantages over more conventional architectures, including parallel processing and reduction in data exchange between memory and the processing units. With that particular limitation in mind, traditional neural network accelerators have been complemented by the use of memristive CNN accelerators. Traditional accelerators comprising GPUs and FPGAs are not meeting the rigid power and latency demands of edge devices. For such plausible edge operations, the memristive arrays provide superior endurance and retention. However, efficient optimization techniques are still required for optimizing CNNs in edge intelligence; hence, various compression techniques, specifically the methods of quantization, have been quite promising in balancing performance with hardware constraints. Lighter CNN architectures such as MobileNet and SqueezeNet further reduce the computational burden, hence making CNNs more adaptable to an edge environment. Dynamic tasks such as speech recognition and video analysis require recurrent neural networks such as LSTM. Implementation of LSTM networks using memristor arrays is going to pose its own problems: how to manage the nonideal effects, for example, and how to efficiently update the weights during training.

Islam et al. [36] discusses similar studies on Convolutional Neural Networks (CNN)-based brain tumor detection. This highlights the importance of CNN in medical imaging including MRI image and to automate tumor detection on MRI images which is more accurate than manual systems. Other studies by Asma Naseer et al., looked at highly accurate CNN models for brain tumor classification. Dataset limitations along with model overfitting and classification errors were encountered as challenges in studies. Some other strategies, for instance, hybrid models and transfer learning were also investigated. Deepak and Ameer used GoogleNet for MRI trait

extraction, with traditional classifiers for higher accuracy [38]. As we can see from the above discussion, CNN model performed much better than any other methods like SVM or DNN especially in medical image processing. In this paper, I extend these results by proposing an improved CNN model for binary classification in the problem of brain tumor detection based on a few-shot learning approach matching high accuracy achieved by best-known models but at a much lower time complexity.

In this paper, Brain Tumor Classification based on Magnetic Resonance Imaging (MRI), Rasool et al. [39] is proposed whereas "A Hybrid Deep Learning Model" can be used for the classification of brain tumors. It is a kind of hybrid model as it uses google-net, a convolution neural network (CNN) pretrained on some data and then two different classifiers — support vector machine (SVM) and softmax respectively. The dataset which we are using consists of the images for 3 types of tumor Glioma, Meningioma, Pituitary tumor and the normal brain image. The hybrid model showed better performance than the other approaches. Here are the accuracies of Google-Net with SVM: The two lists Lth and Th were exactly 98.1% as compared to the two approaches that used fine-tuned models, which achieved between 0.3 and 93%Ẇe found that SVM produced relative improvements in classification with the operation of introducing regions, which demonstrated the good characteristic pattern-recognition ability of it. Here, we propose a model for automating the classification of tumors that assist radiologists to avoid human error in tumor detection and this should be a potential diagnostic tool for early operation before diseases become invasive.

Author Walsh et al. [40] in this paper reviews different methods to segment brain tumors from MR images. That means, there will be a transition from ordinary machine learning methods based on manual feature engineering to those deep learning approaches, especially based on CNNs, that experience huge developments. This paper places special focus on a popular deep learning architecture, U-Net, which is well-known for generating high-quality segmentations with much less training data, and without the use of complex image augmentation strategies. It further describes other architectures such as MRNet, historical methods Thresholding and clustering algorithms. One of the key gaps identified was a lightweight and effective segmentation model. Although previous models are performant, they usually utilize much computation or huge numbers of annotated data. The paper sets its proposed lightweight U-Net as an amelioration that achieves real-time, accurate segmentations while having significantly less computational requirements enabling it to be utilized in practical clinical usage.

In this paper, author Khan et al. [47] includes various types of research based on how quantum machine learning Quantum applications have advanced into medical imaging especially in the segment dedicated towards Brain tumor classification. It cites previous work showing that the hybrid quantum-classical models such as HQC-CNNs could outperform classical ones in tumor classification [26]. Parallel to these efforts, additional research looked into quantum models designed to enhance the encryption techniques needed for processing confidential and accurate medical data. Quantum computing has been combined with several methodologies to address drawbacks such as overfitting and expenses in computational resources associated

with classical CNNs, thereby enriching models for quality and cost efficiency. Additionally, his work illustrated how quantum neural networks can help tackle the increasing complexity of medical datasets and real-world use cases such as pediatric brain tumor classification and segmentation. Based on the above foundation, this paper presents a QCNN network by combining traditional CNN blocks and experiments prove it's significantly significant in accuracy and calculation efficiency.

In this paper, author Alzahrani et al. [46], introduced a new tutor–student approach to the detection of brain tumors by multiclass, using transformer architectures with tripartite attention mechanisms based on distillation. The proposed approach extends MRI-based brain tumor classification with the addition of a few neighborhoods' attention, global attention, and cross-attention transformer layers. This enables compression of the knowledge contained in larger, slower Teacher models into smaller and faster Student models, which are deployed on resource-constrained IoT devices, referred to for convenience as the Internet of Medical Things-IoMT. This model has been verified based on the highly augmented MRI data, which covers both local and global contextual information and hence should enhance the feature representation of brain tumor types. The performance of the tripartite attention mechanism is evidenced in the study, with results showing a higher accuracy rate than traditional CNNs and other transformer-based models. The results of this study provide promising insights into accuracy of classification, Brier scores, and AUC, therefore representing an important advance in AI-aided medical diagnosis, especially in real-time or low-resource detection of brain tumors.

In the following paper, author Geile et al. [49] elaborates on the efficient method for detection and classification of brain tumors with the help of pre-trained CNN models. The authors introduce ResNet50, EfficientNet, and two deep learning models for classifying MRI images of brain tumors. To enhance model performance, the technical tool Transfer learning and Data augmentation were applied. These methods tackle the data paucity issue by employing pretrained models on generic, large-scale datasets, and augmenting existing data libraries through transform operations. The results showed that ResNet50 and EfficientNet models had better classification accuracy. As we can observe, the ResNet50 performs with an accuracy of 96% in testing and the EfficientNet does a bit better getting to 98%Ḋata augmentation improved the validation accuracy from 71.3% to (around) 80%Ẇhile this study raises the importance of CNNs within medical image analysis, there are areas which could be strengthened such as improving data availability and possibly other features in newer architectures.

In this paper, author Fu et al. [23] puts forward an idea of improving the perception capability of vehicular networks by utilizing camera and millimeter-wave radar through MEC. The main objective is to overcome the shortcomings of the conventional in-car sensors especially when it comes to managing long distances, intersections, and other forms of interference. They then introduced a deep learning algorithm-based fusion method for the analysis of YOLOv3 camera data, while the DBSCAN clustering algorithm forms part of the analysis in radar data. Multi-sensor integration of the signal data and synchronization both in space and time domain are also used in the method together with the Munkres algorithm on object association

and Kalman filtering for multi-object tracking. The fusion method applied on the MEC platform also provides low-latency, high-accuracy perception that is particularly important for real-time applications such as V2X. Simulation outcomes prove that by applying the proposed method, object detection achieves better stability and accuracy when compared to separate usage of both cameras or radar. This work is useful in the field by proving the usefulness of edge computing in improving the sensor fusion in vehicular networks hence the safety and efficiency of self-driving cars.

Author Hamanaka et al. [30] introduces an economic and healthy way of achieving the autonomous driving systems on the edge devices by using the Binarized Neural Network (BNN) accelerator on FPGA. It covers positions in loading deep neural networks on resource-constrained devices through the introduction of BNN and, similarly, BNNs that will virtually eliminate parameter size and computational latency. Next to wavering Raspberry Pi-based alternatives with which its latency was 236 inferences per second, this system is developed out of them in the first place and demonstrates top-notch real-time performance. The authors have also refactored the GUINNESS into a tool for BNN to take out the use of off-chip memory and hence increase the system's overall efficiency. The paper emphasizes the educational aspect of this system, as it can be done at lower cost and has a design that is accessible. Though, there are a few shortcomings with the turning of the curves towards the vehicle even in straight-line movement, suggesting the application of more advanced neural networks for duties such as object recognition and driving more complex scenarios.

In this paper, author Matos et al. [48] is about the problem by incorporating neural network equivalence verification into search-based quantization. It iteratively develops the original high-precision network to synthesize a quantized model, which also has the same behavior as the high-precision one. The authors show that CEG4N can achieve significantly higher accuracy QNNs (163 % over baseline). Counterexample Quantization brings a balance between resource savings and model performance by leveraging counterexamples in quantization. It is very helpful in the model deployment under a low-resource context. Experiment results show on agility benchmarks like ACAS Xu, MNIST and CIFAR-10 that the proposed CEG4N has better performance to preserve equivalence between original model and quantized one with minimum precision loss. This contribution addresses the wider area of optimization of neural networks, more specifically in safety-critical domains and presents a general approach for improving the reliability of quantized models.

Author Nguyen et al. [14] in this paper, describes the simulation framework developed to perform performance, energy efficiency, and communication strategy evaluations inside ECC environments. ECSim++ is based on OMNeT++ and INET; hence, users can simulate cloud services at the network edge to reduce latency and backhaul traffic. Unlike simulation tools available so far, such as CloudSim and iFogSim, the edge-specific features supported in ECSim++ include service management, energy consumption analysis, and caching strategies. The tool embeds a novel Enoma protocol for the management of services across edge nodes with the aim of enhancing resource allocation and energy efficiency. ECSim++ further enhances large-scale simulations by making the architecture modular; it thereby provides an

efficient way to model, test, and optimize ECC solutions, focusing on energy-saving strategies and service delay optimization. The paper goes further to advance the simulation tools for ECC to bridge the gap left by earlier platforms and provides a comprehensive environment for edge cloud research.

Liang et al. [3], in this paper, has discussed the detection of security vulnerabilities both in source code and binary executables. It has pointed out that source code analysis techniques are based upon static analysis; hence, these techniques result in a large number of false positives due to the absence of runtime information. On the other hand, binary executable dynamic analysis methods have problems in terms of efficiency due to test case generation. In light of this, the authors propose a collaborative analysis method that unifies source code and binary executable analysis using a uniform defect mode set. The approach is, therefore, meant to push forward the accuracy and performance in vulnerability detection, overcoming some of the limitations related to both static and dynamic analyses. This approach ensures that the defects activated in runtime of the binary executables are attributed to the ones found in source code through a common framework. A validation of this approach is done in the paper by an experimental example showing how effective collaborative analysis can be in detecting software security vulnerabilities.

Paper (Author, Year)	**Objective/ Focus**	**Methodology**	**Key Findings**	**Gaps / Limitations**
Rastegari et al., 2016	Faster computations via weight binarization	Binary-Weight Networks (BWN) and XNOR-Nets	Reduced memory /computation with minimal accuracy loss	Focused on image tasks; lacks application in NLP and video
Bulat et al., 2017	Improvement of BNNs' Predictive Performance on Complicated Datasets	XNOR-Net++ with combined scaling factors for weights and activations, tested on ResNet18 and AlexNet using ImageNet	XNOR-Net++ significantly improved accuracy for BNNs on ImageNet, with Top-1 accuracy of 57.1% on ResNet18	BNNs still struggle to match full-precision networks on very complex tasks; modest improvements on shallow networks like AlexNet
Helwegen et al., 2020	Training BNNs from scratch without full-precision pretraining	Modified ResNet (ResNetE) and DenseNet (DenseNetE) architectures evaluated on MNIST, CIFAR-10, and ImageNet	ResNetE-18 achieved 87.6% on CIFAR-10 and 56.7% on ImageNet; DenseNetE-21 achieved 90.3% on CIFAR-10 and 57.1% on ImageNet	No weight decay used; scaling factors deemed unnecessary; struggles with extreme accuracy compared to full-precision models

Table 2.1: Summary of Reviewed Papers and Identified Gaps

Chapter 3

Experimental Setup

3.1 General Outline

For the research purpose, we have applied binarization on several CNN models such as: VGG16, ResNet18, DenseNet121. The mentioned models were selected because of its performance and the effectiveness of binarization. For dataset, we used a restricted dataset from the Cancer Imaging Archive. A general work plan diagram for our proposed binarized system in given in Figure 3.1.

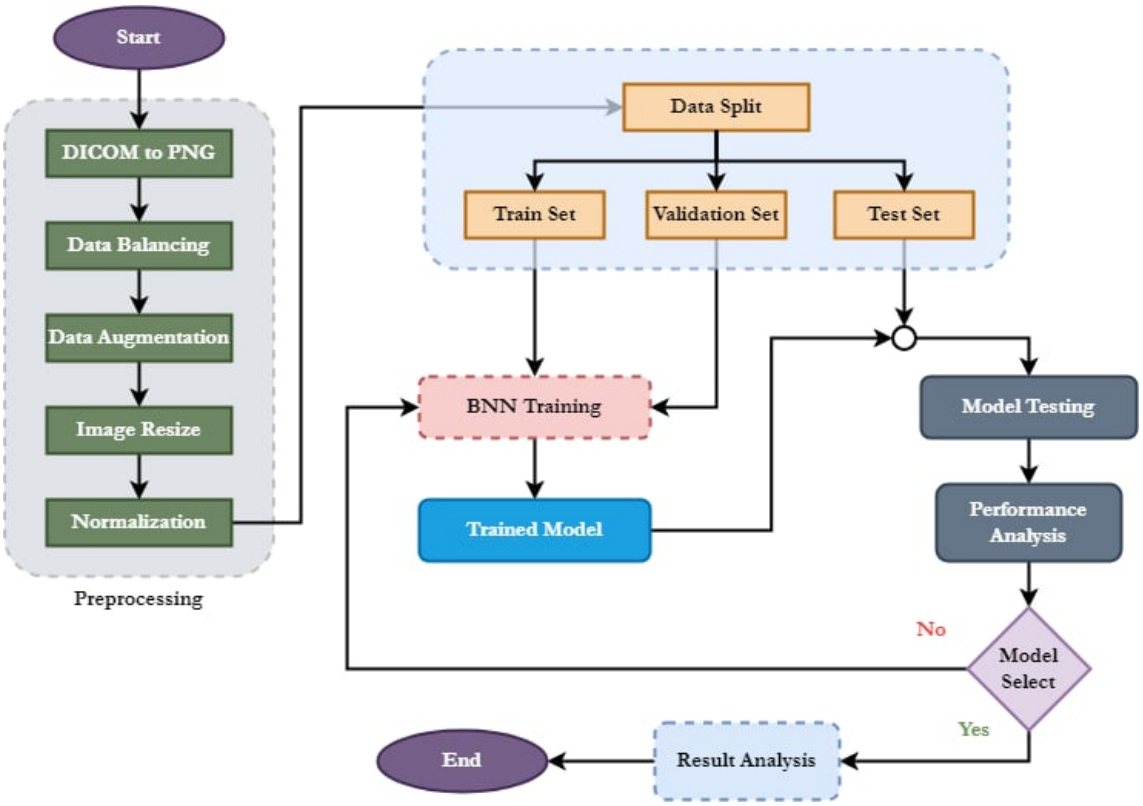

Figure 3.1: Workflow of proposed BNN

3.2 Platform and Language

The novel model idea Binary Neural Network (BNN) for cancer detection The development of the proposed BNN Model to detect Glioblastoma was implemented in Python Throughout this paper. Python was chosen due to its flexibility and being one of the most well-liked languages in machine learning but also has excellent support for deep neural network frameworks. The model itself is implemented with the PyTorch deep learning library, which makes it easy to create binary-weight networks. It used helper libraries like NumPy for numerical calculations, Matplotlib to draw results and Scikit-socket or data preprocessing as well as evaluation matrices. We developed the environment on a high-performance PC with AMD Ryzen 9 5950X Processor, 64GB of DDR4 RAM, and NVIDIA RTX 3080Ti GPU, with 12GB of dedicated video memory. Running on Windows 10, this system was able to fall on speed and storage capacity to sustain the speed necessary for large datasets and train the BNN model without performance bottlenecks. The result is this configuration, which enabled smooth deep learning experiment execution and rapid model iterations.

3.3 Dataset Description

The Cancer Genome Atlas Glioblastoma Multiforme (TCGA-GBM) [7] dataset used for the Glioma class was obtained from The Cancer Imaging Archive. The genomic data in this dataset includes the mutation counts, gene expression profiles, copy number variations, and methylation patterns. The clinical data includes disease status, treatment responses, and clinical outcomes. By combining the clinical data with genomic data, we can get accurate differences between lower-grade gliomas(LGG) and glioblastomas(GBM), which we can then use to detect gliomas. This dataset comprises of T1-weighted, post-contrast T1-weighted, T2-weighted, and fluid-attenuated inversion recovery (FLAIR) sequences. Those image samples were taken along with Siemens, Philips, and GE Medical Systems scanners.

The Information eXtraction from Images (IXI) [1] dataset is used for No-tumor class. This dataset has T1-weighted, T1-weighted with gadolinium contrast enhancement (T1ce), T2-weighted, as well as T2-weighted Fluid Attenuated Inversion Recovery (FLAIR) sequences for glioblastomas and low-grade gliomas (LGG), as well as MRI scans from healthy patients with no-tumor in the brain. Fig 1 shows MRI samples from (TCGA-GBM) [7], and Fig 1 shows MRI samples from The Information eXtraction from Images (IXI) dataset [1].

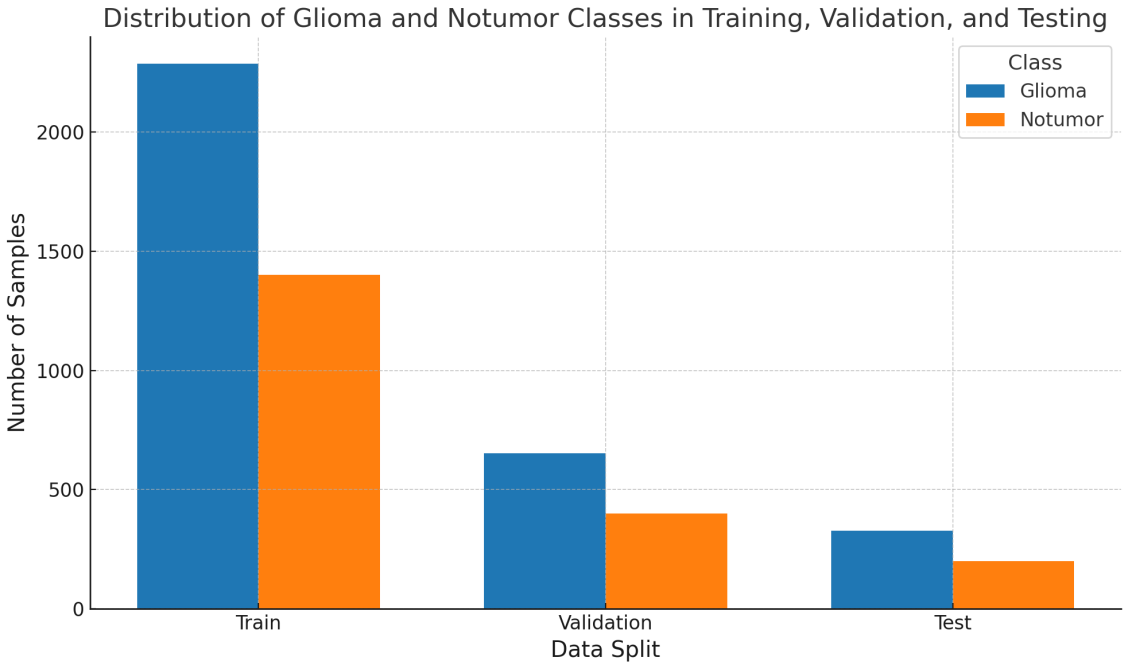

Figure 3.2: Distribution of Glioma and Notumor classes in Training, Validation, and Testing

3.3.1 Dataset Overview

For our research, we used The Cancer Genome Atlas Glioblastoma Multiforme (TCGA-GBM) dataset [7] for Glioma class and The Information eXtraction from Images (IXI) [1] for No-tumor class. There are a total of five thousand two hundred sixty-six samples merged in 2 classes. The dataset is slightly imbalanced, which we handle in the data preprocessing section. Fig. 3.3 represents MRI samples from each class in the dataset.

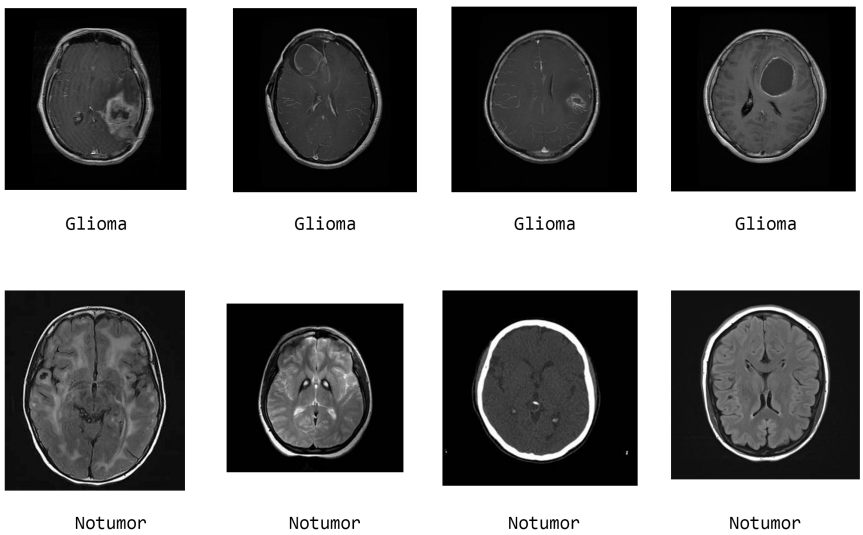

Figure 3.3: Sample of MRI images from the TCGA-GBM and IXI dataset.

Chapter 4

Methodology

4.1 Binary neural networks

When a Neural Network contains binarized activations and weights varying from -1 to +1 across all layers, it is known as a Binary Neural Network. As binarization reduces computation from a 32 bit floating point operation to a bitwise operation, a drastic change takes place in memory usage and computational costs. The key factor that makes BNN an efficient alternative over traditional CNN is the lower memory usage with 32x faster inference speed for embedded systems [45]. Our research delves deeper into this network structure to find and provide optimal solutions based on the modern world's real time scenario.

As discussed earlier, BNNs have binarized activations and weights that cause huge degradation in accuracy during the training process, which accelerates toward loss of information. To minimize the dissipation, we adopted a strategy known as Binary Weighted Neural Network that uses binary weights instead of full precision weights to allow for execution on mobile devices by speeding up inference [10]. But this idea is further modified where we only use binarized weights during forward pass, keeping the activations and layer calculation in full precision. Now, this idea will be implemented to binarize CNN models like - ResNet18, DenseNet121 and VGG16 to achieve our goal of building a lightweight model.

4.2 Data Preprocessing

Our TCGA-GBM dataset [7] contains MRI data in DICOM format. So our preprocessing starts with extracting the images from the DICOM format. That is to mention, the DICOM format includes MR images and metadata as in patients' information. However, for our brain tumor detection task, only image data was necessary. In order to extract the images only, first we used a pydicom library that stores the pixel arrays of the images. After storing the pixel array information, we scaled the pixel values and converted them into PNG format. When the conversion is done, then resizing the input images into 224x224 pixels was handled using nearest-neighbor interpolation, after that they were normalized. Next is dataset splitting. We divided our dataset into three parts - training, validation, and testing

set in 7:2:1 ratio. We have implemented data augmentation on the training set for a better model performance to add variation and diversity on the input data. Data augmentation includes horizontal random flip, random rotation (20 degree), color adjustment - brightness, contrast and saturation by a factor 0.2, random shear/tilt (15 degree). Lastly, data balancing was done by oversampling the no tumor class of the IXI dataset [1] .

4.3 Description of the model:

4.3.1 BWN ResNet18:

ResNet18 is a deep convolutional neural network, consisting of 18 layers, and is designed to address the vanishing gradient problem that often hinders the training of very deep networks. It achieves a solution by introducing residual learning, allowing gradients to proceed through the network easily, even if the number of layers expands.

ResNet18 consists of 18 layers structured as follows: it begins with an initial convolution layer using 7x7 convolutions with 64 filters and a stride of 2, after that, is batch normalization and ReLU activation. Consequently comes a max pooling layer with a 3x3 kernel and a stride of 2. The core of ResNet18 is comprehends four residual blocks, where each block accommodates two convolution layers with 3x3 kernels, batch normalization layers, and ReLU activations. These blocks incorporate an identity shortcut connection, which enables the input to bypass the convolutional layers and be added to the output of the block. This process is the key to the residual learning mechanism, where each residual block can be represented mathematically using:

$$\boldsymbol{y} = F(\boldsymbol{x}, \{\boldsymbol{W_i}\}) + \boldsymbol{x} \tag{4.3.1}$$

where,

- $\boldsymbol{x}$ is the input to the residual block.
- $F(\boldsymbol{x}, \{\boldsymbol{W_i}\})$ represents the residual mapping to be learned.
- $\boldsymbol{y}$ is the output of the block.

In these residual blocks, function F incorporates two 3x3 convolution layers with batch normalization and ReLU activation, introducing non-linearity. The architecture concludes with an average pooling layer and a fully connected layer.

To modify ResNet18 for binary weight calculation, we introduce several key changes: At the very beginning, the first convolution layer is modified to accept single-channel grayscale images. Next modification step includes binarizing weights of the convolution and the fully connected layers during forward pass. But before binarizing the weights we need to calculate the scaling factor α, which is the absolute mean value of real weight $\boldsymbol{W}$. Mathematically it can be written as:

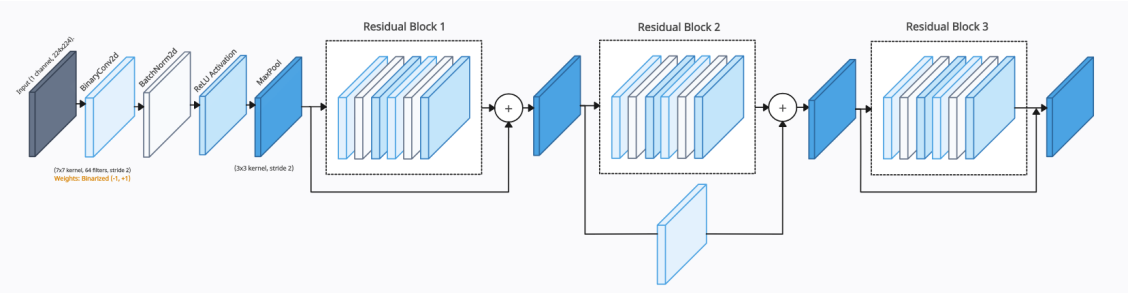

Figure 4.1: Binarized ResNet18 Model architecture

$$\boldsymbol{\alpha} = \boldsymbol{E}(|\boldsymbol{W}|) \tag{4.3.2}$$

After computing the scaling factor $\boldsymbol{\alpha}$, the weights are binarized using sign function. So for each layer with weight W is binarized as:

$$\mathbf{W_b} = \alpha \cdot \text{sign}(\mathbf{W}) \tag{4.3.3}$$

where, $\boldsymbol{W_b}$ is the binarized weight

and $sign(\boldsymbol{W})$ is the sign function:

$$\text{sign}\,(\boldsymbol{W}) = \begin{cases} 1 & \text{if } \boldsymbol{W} \geq 0 \\ -1 & \text{if } \boldsymbol{W} < 0 \end{cases}$$

A recursive function systematically replaces all the convolutional and linear layers in the network with their binary versions, ensuring that all computations are performed using binarized weights. Afterward, as we calculate the binarized weight, next it's used for the layer's computation. So the layer's output becomes:

$$\boldsymbol{y} = \sigma\left(\text{BN}\left(\boldsymbol{W_b} \cdot \boldsymbol{x}\right)\right) \tag{4.3.4}$$

where,

- $\boldsymbol{x}$ is the input.
- $\boldsymbol{W_b}$ is the binarized weight.
- BN is the Batch Normalization.
- σ is the activation fuction (ReLU in this case).

When the model is done with a forward pass, it generates an output. With this output, the model performs an update which, backpropagating, calculates the loss L on that output using a straight-through estimator to estimate gradients with respect to full precision weights via the sign function. Loss function L can be expressed as:

$$\frac{\partial L}{\partial W} \approx \frac{\partial L}{\partial W_b} \cdot \frac{\partial W_b}{\partial W} \approx \frac{\partial L}{\partial W_b} \tag{4.3.5}$$

This operation allows the gradients to flow through the network enabling it to learn in spite of the non-differentiable sign function.

The modified ResNet18 (BWN ResNet18) architecture maintains its residual learning structure but with binarized weights at each stage, allowing the network to efficiently process input images while maintaining feature extraction capabilities, reducing model size, computation requirements and speeding up operations which are crucial for detecting glioblastoma. Furthermore, during the backward pass, gradients are computed with respect to full-precision weights, which enables the network to learn effectively. However, the binarization process can lead to information loss and a relative reduction in model performance compared to the standard version.

4.3.2 Binarized DenseNet121:

DenseNet121 is a special architecture where each layer in the network takes input from all the preceding layers. The design allows for the reusing of features, which acts as a solution to the vanishing gradient problem and optimizes gradient flow during backpropagation. The architecture is made up of dense blocks, for every connection (i.e., layer) within a block each other connects in the feedforward way. This means that these transition layers are mostly used between dense blocks, which can reduce feature maps through convolution & pooling operations. The architecture of the model contains an input layer in addition to four dense blocks containing different depths (6, 12, 24 and 16 layers) and a classification layer using global average pooling followed by a fully connected output network. We chose 32 for the growth rate k, which defines the number of filters added at each layer and influences model efficiency in feature representation.

Deploying DenseNet121 on binary weight networks (BWN) involves various changes to accommodate this model. In the beginning, all convolutional layers were replaced with BinaryConv2d layers to allow forwarding through binary weights only and reducing memory consumption as well as computational complexity drastically. Finally, the linear classifier layer is replaced with a BinaryLinear to ensure that weight binarization behaves as before throughout the model. However, scaling factors (α) are projected to keep the lost magnitude information after binarization. Further modification is done because the images are grayscale; thus, the first convolutional layer is altered to accept 1-channel input instead of the common 3-channel RGB images. Adaptations along this line make the model effective, improving its practicability for real-world applications.

The mathematical bases of BWN involve binarization techniques that transform real-valued weight tensors W into binary representations. These binary weights are then used by the convolution operations in a forward pass to carry out fast computations. This is followed by the application of an activation function to introduce

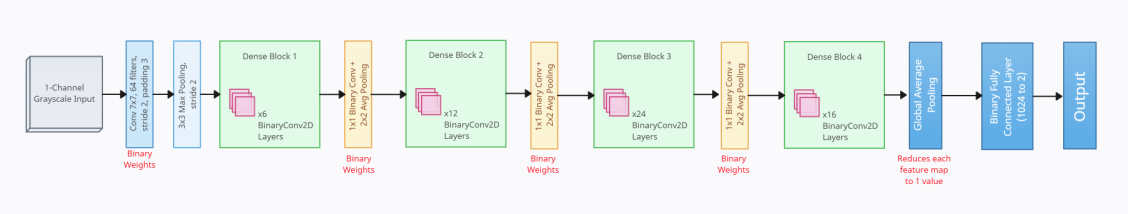

Figure 4.2: Binarized DenseNet121 Model architecture

non-linearity into the model. It maintains dense connectivity by concatenating all previous layers' outputs to the current layer's output. This would lead to richer feature representation and resolve some issues of learning dynamics.

Weight Binarization

For a real-valued weight tensor W:

1. Compute Scaling Factor (α):

$$\alpha = \frac{\mathbf{1}}{\mathbf{n}} \sum_{\mathbf{i=1}}^{\mathbf{n}} (|\mathbf{W_i}|) \tag{4.3.6}$$

   where $\mathbf{n}$ is the number of elements in $\mathbf{W}$.

2. Binary Weights $\mathbf{W_b}$:

$$\mathbf{W_b} = \alpha \cdot \text{sign}(\mathbf{W}) \tag{4.3.7}$$

$$\text{where} \quad \text{sign}(\mathbf{W}) = \begin{cases} +1, & \text{if } W \geq 0 \\ -1, & \text{if } W < 0. \end{cases}$$

Forward Pass with Binary Weights

1. Convolution Operation:

$$\mathbf{Y} = \mathbf{W_b} * \mathbf{X} + \mathbf{b} \tag{4.3.8}$$

   where $*$ denotes convolution, $\mathbf{X}$ is the input, and $\mathbf{b}$ is the bias term.

2. Activation Function:

$$\mathbf{Z} = \mathbf{ReLU}(\mathbf{Y}) = \mathbf{max}(\mathbf{0}, \mathbf{Y}) \tag{4.3.9}$$

Dense Connectivity

1. Concatenation: Each layer's output is concatenated with all preceding layers' outputs:

$$\mathbf{X^{(l)}} = \left[\mathbf{X^0}, \mathbf{X^1}, \ldots, \mathbf{X^{(l-1)}}\right] \tag{4.3.10}$$

   where $\mathbf{X^{(l)}}$ is the input to layer l.

The architecture of DenseNet121 includes some key components: the input layer, dense blocks, transition layers, and final layers. It starts with the input layer performing the first convolution using binary weights to be computationally efficient. Within the Dense Blocks, a number of dense layers are employed, starting with a Bottleneck Layer that first reduces the dimensions by the use of a 1×1 convolution and then employs a Convolutional Layer to extract features using a 3×3 convolution. The Transition Layers reduce the size of feature maps and the number of feature maps between the dense blocks; binary convolutions together with average pooling for further efficiency are used. Lastly, the final layers comprise of batch normalization and activation functions, culminating in global average pooling. This is then followed by a binary linear layer with SoftMax activation, which predicts class probabilities, meaning that the model can therefore make informed predictions using the learned features.

The input images are first fed into the initial convolution and pooling stage of the BWN DenseNet121 model, afterwards, they pass through four binary weighted dense blocks. Dense connectivity allows each block to reuse features, while transition layers reduce dimensionality efficiently. Lastly, the classification layer uses global average pooling and a binary linear layer, where output values are fed through the softmax activation function to produce class probabilities.

4.3.3 Binarized VGG16:

Among the most interesting convolutional neural networks, the architecture of VGG16 is famous due to its simplicity and depth. This model is reduced to a set of key components that altogether work in an effective way to classify images. The small receptive field of size 3x3 in the Convolutional Layers 13 may enable the model to catch even finer details from the input images. These are followed by a total of 5 Max Pooling Layers, which reduce the dimensions of the feature maps, maintaining only the most useful information. The architecture also consists of 3 Fully Connected Layers, which are basically classifiers that would be transforming the extracted. This takes features to class probabilities. A core component of this model is using a Rectified Linear Unit (ReLU) activation adding non-linearity to this model and enhancing its capability to learn difficult patterns. VGG16 has established itself as a robust and effective model, widely adopted for different image classification tasks,

which includes successive convolution and pooling layers for feature extraction and fully connected layers for classification.

The BWN VGG16 model represents a modification of the standard VGG16 architecture by introducing binary weights according to the Binary Weight Networks approach. This network is different, as it replaces all conventional convolutional layers with BinaryConv2d layers for efficient weight binarization. All linear layers are replaced by BinaryLinear layers to keep consistency in the form of the weight representation. It has been modified to take one-channel grayscale images as input instead of the traditional three-channel RGB format that can be used to process MRI images. Scaling factors (α) are used to counteract performance degradation due to weight binarization and preserve the model's representational capacity.
We will go through layer by layer, identifying the mathematical operation and terminologies related to the architecture of the BWN VGG16 model. The Input Layer accepts grayscale images with minimal preprocessing, resized to 224 $\times$ 224 pixels. Mathematically this can be denoted as:

$$X \in \mathbb{R}^{1 \times \boldsymbol{H} \times \boldsymbol{W}} \tag{4.3.11}$$

where $\boldsymbol{H}$ and $\boldsymbol{W}$ are the height and width of the image.

Among the Convolutional Layers with Binary weights, each one of these layers performs several key operations.

These are Weight Binarization, which is a scaling factor α that transforms the real weights into binary weights. So, first the binarization process will start from taking the input image and doing matrix multiplication with the kernel, which produces an output feature map with weight W. Mathematically it can be represented as:

$$\mathbf{W} \in \mathbb{R}^{\mathbf{C_{out}} \times \mathbf{C_{in}} \times \mathbf{k_h} \times \mathbf{k_w}} \tag{4.3.12}$$

where $\mathbf{n} = \mathbf{C_{in}} \times \mathbf{k_h} \times \mathbf{k_w}$ and $\boldsymbol{w_i}$ are elements of $\mathbf{W}$.
Before converting the real weight matrix to binarized weight, we need to calculate a scaling factor α. Mathematical representation is:

$$\alpha = \frac{\mathbf{1}}{\mathbf{n}} \sum_{\mathbf{i=1}}^{\mathbf{n}} (|\mathbf{w_i}|) \tag{4.3.13}$$

Then the weight matrix will be multiplied with the scaling factor and produce binary weights. Which is:

$$\mathbf{W_b} = \alpha \cdot \text{sign}(\mathbf{W}) \tag{4.3.14}$$

where $\text{sign}(\mathbf{W}) = \begin{cases} +1, & \text{if } W \geq 0 \\ -1, & \text{if } W < 0 \end{cases}$

After binarizing the weights, output of each convolutional layer is determined by:

$$\mathbf{Y} = \mathbf{W_b} * \mathbf{X} + \mathbf{b} \tag{4.3.15}$$

where $*$ denotes the convolution operation, $\mathbf{b}$ is the bias term (if used), and $\mathbf{Y}$. is the output feature map.

Following this, the Activation Function applies ReLU activation that introduces non-linearity into the model, defined as:

$$\mathbf{Z} = \mathbf{ReLU}(\mathbf{Y}) = \mathbf{max}(\mathbf{0}, \mathbf{Y}) \tag{4.3.16}$$

The Max Pooling in the Pooling Layers reduces the spatial dimensions without losing any important features.

$$\mathbf{P} = \text{MaxPool}(\mathbf{Z}) \tag{4.3.17}$$

Typically uses a **2x2** window with stride 2.

In the Fully Connected Layers with Binary Weights, the output from the last convolutional layer is flattened into a vector,

$$x \in \mathbb{R}^N \tag{4.3.18}$$

Then Weight binarization is again applied to these fully connected layers in similar procedures as before, which is represented as:

$$\boldsymbol{W} \in \mathbb{R}^{\boldsymbol{N_{out}} \times \boldsymbol{N_{in}}} \tag{4.3.19}$$

Like before, scaling factor α is calculated using the formula:

$$\alpha = \frac{1}{N_{in}} \sum_{i=1}^{N_{in}} |w_i| \tag{4.3.20}$$

Then the scaling factor is multiplied with the sign function of the weight matrix.

$$\mathbf{W_b} = \alpha.\mathbf{sign}(\mathbf{W}) \tag{4.3.21}$$

the linear transformation is denoted as follows:

$$\mathbf{Y} = \mathbf{W_b} \cdot X + \mathbf{b} \tag{4.3.22}$$

It is important to note that ReLU activation is applied except in the final output layer.

$$\mathbf{Y_{out}} = \mathbf{max}(\mathbf{0}, \mathbf{Y}) \tag{4.3.23}$$

$$\mathbf{z} = \mathbf{ReLU}(\mathbf{y}) \tag{4.3.24}$$

Finally, the SoftMax output layer generates predictions for class probabilities.

$$\hat{\boldsymbol{y}} = \text{Softmax}(\boldsymbol{y}) = \frac{\exp(\boldsymbol{y})}{\sum_k \exp(\boldsymbol{y_k})} \tag{4.3.25}$$

Special Cases: The pooling layers do not involve weights or activation functions; their outputs are directly fed into subsequent layers.

$$\boldsymbol{X^{(l)}} = \boldsymbol{Y^{(l)}} \tag{4.3.26}$$

Instead of ReLU, softmax is used in the final layer to provide class probabilities.

$$\hat{\boldsymbol{y}} = \text{Softmax}(\boldsymbol{Y^{(L)}}) \tag{4.3.27}$$

Cross-Entropy Loss applied for multi-class classification,

$$\mathcal{L} = -\sum_{i=1}^{C} y_i \log(\hat{\boldsymbol{y}}_{\boldsymbol{i}}) \tag{4.3.28}$$

where $\boldsymbol{y_i}$ is the true label (one-hot encoded), and $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$ is the predicted probability for class $\boldsymbol{i}$.

For backpropagation with binary weights,

We compute the gradients using the real valued weights, W, rather than the binary weights, W_b, during the backpropagation. This ensures that the model has truly learned from the underlying distribution of the data.

A common approach in this regard is the Straight-Through-Estimator (STE), which approximates the gradient of the sign function and allows gradients to flow through binary activations. This improves the training methodology for Binary Neural Networks, which can now be efficiently optimized without losing performance.

$$\frac{\partial \operatorname{sign}(x)}{\partial x} \approx \begin{cases} 1, & \text{if } |x| \leq 1 \\ 0, & \text{otherwise} \end{cases} \tag{4.3.29}$$

Herein are some basic definitions and explanations of important mathematical terms used in this work. Let X represent the input or intermediate feature maps processed by a network. Let W be the real-valued weights learned during training and let W_b be the binary weights resulting from a binarization process. The symbol α represents the scaling factor, which is the average of the absolute values of the weights, used to regularize the scale in order not to hurt the performance of the model. The term b is the bias at each layer, increasing the ability of the model for flexibility in learning. refers to the convolution across feature maps. ReLU is the Rectified Linear Unit, an activation function that introduces non-linearity into the model. MaxPool reduces the spatial dimensions with no loss of critical features. y is the predicted probabilities after applying SoftMax, L refers to the loss function which assesses the model's performance, and lastly, the sign function returning +1 or -1 is represented as sign(x) crucial for the weight binarization process.

BWN VGG16 was designed to operate on specific inputs for a certain purpose efficiently. Input Size: The network expects the input images to be resized to 224 × 224 pixels, which provides consistency in the dataset. Gray-Scale Images: Since MRI images are inherently grayscale, the input layer has been modified to take the image with one channel instead of the usual three-channel; this becomes necessary when dealing with medical imaging data. Class Labels: In the case of binary classification, where the classes are "Glioma" versus "No Tumor," the output layer consists of two neurons. This architecture will enable the model to give a probability estimate for each class; it will, therefore, help in making efficient decisions regarding the detection of glioblastoma.

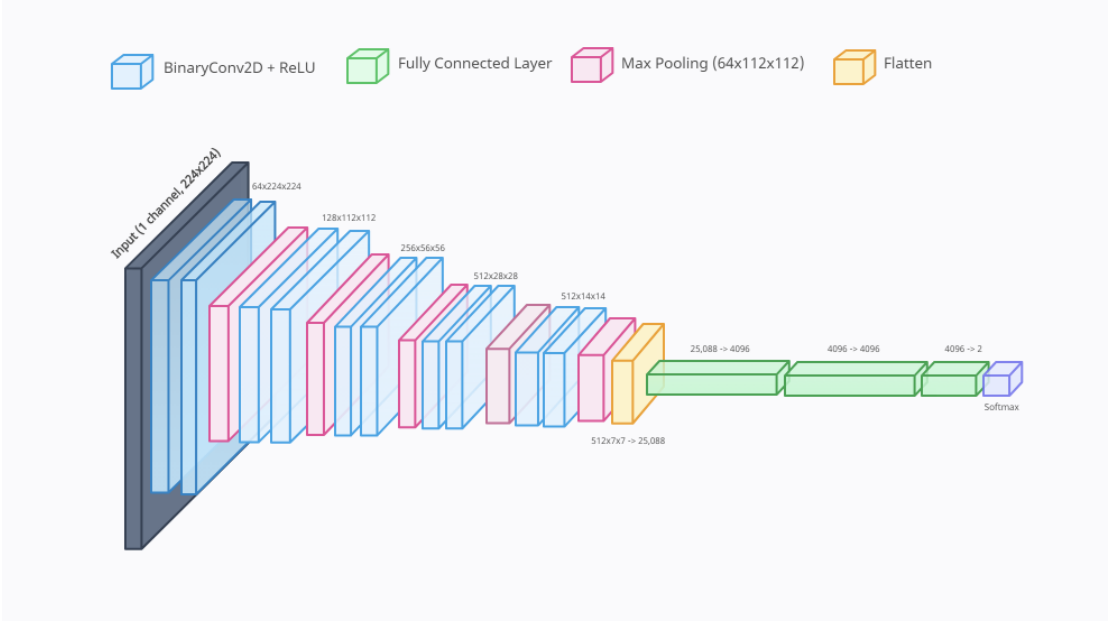

Figure 4.3: Binarized VGG16 Model architecture

Chapter 5

Result Analysis

5.1 Quantitative Performance Metrics

5.1.1 BWN ResNet18

After finishing the training of the Binary Neural Network (BNN) model on the acquired glioma image dataset, we evaluated the performance of this model based on a number of quantitative metrics (including accuracy, precision, recall, F1-score and the ROC-AUC curve) on the test set.

We have trained this model for 60 epochs with a learning rate of 0.01, momentum of 0.9 and learning rate decay settings at 30 and 40 with a gamma factor of 0.01 and of batch size of 64. On the test set the final model's overall accuracy was 85.01% .

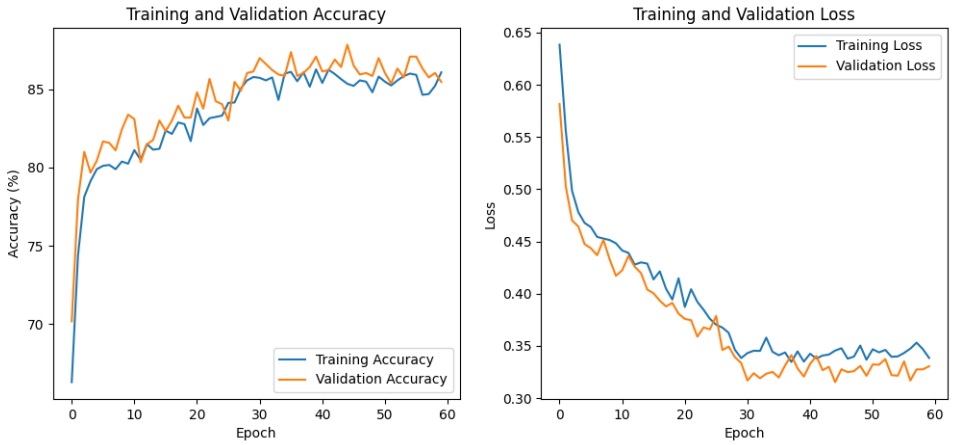

Figure 5.1: Accuracy & Loss Curve of ResNet18's Training and Validation Data

The training and validation accuracy along with the loss curves are plotted in Figure 5.1. It is noteworthy that the validation accuracy outperforms training accuracy and reached 89% after 60 epochs. Loss plots show that both the training and validation loss are reduced consistently through the epochs and the validation loss ends up at around 0.33 at the last epoch, indicating minimal overfitting.

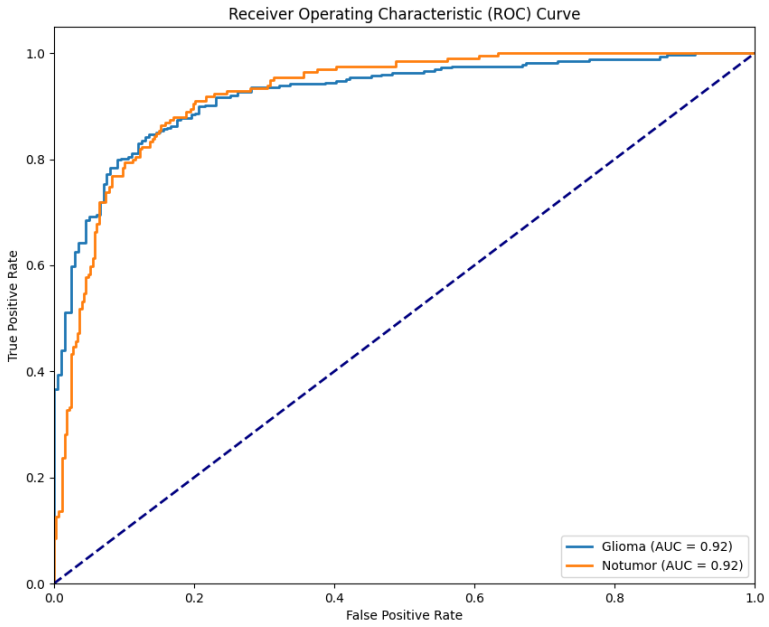

Figure 5.2: ROC Curve of ResNet18

Additionally, as illustrated in the ROC curve in Figure 5.2 , the model's Area Under the Curve (AUC) score was 0.9229. This high AUC score demonstrates the model's ability to accurately distinguish between the two classes: 'glioma' and 'notumor.' The closer the AUC score is to 1, the better the model is at making correct classifications. In this case, an AUC of 0.9229 indicates that the model performs exceptionally well, with a strong capacity to separate true positives (glioma cases) from true negatives (no tumor cases). The ROC curve thus visually confirms the effectiveness of the ResNet18 model in addressing the binary classification task for glioblastoma detection, highlighting its potential for reliable diagnostic use.

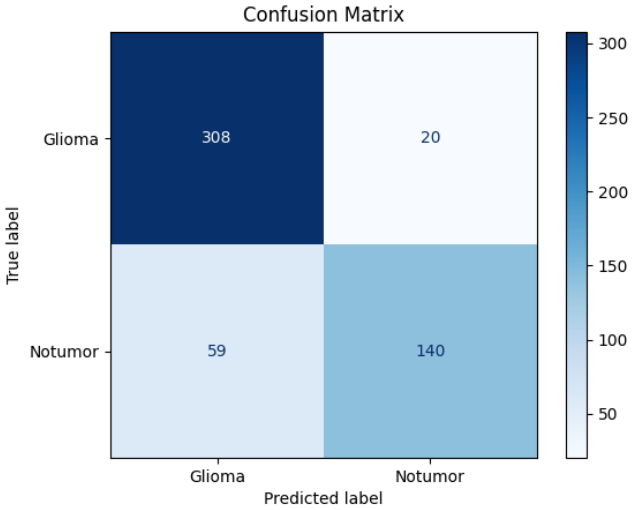

Figure 5.3: Confusion Matrix of ResNet18

The confusion matrix in Figure 5.3 indicates that the model classifies 308 out of 328 glioma images and 140 out of 199 no tumor images for glioma detection and no tumor detection respectively, which is in accordance with the precision and recall previously mentioned.

5.1.2 BWN DenseNet121

This work used a Binarized Neural Network based on the DenseNet121 architecture for glioblastoma detection from MRI images with very high efficiency. This model aims to provide a cost-friendly and lightweight solution suitable for mobile devices, targeting healthcare professionals such as doctors and nurses. This work used a Binarized Neural Network based on the DenseNet121 architecture for glioblastoma detection from MRI images with very high efficiency.This model aims to provide a cost-friendly and lightweight solution suitable for mobile devices, targeting healthcare professionals such as doctors and nurses.

From the training details, a few facts could be deduced regarding the performance of the model. Firstly, the model was trained for 100 epochs with a batch size of 64. Second of all, the learning rate is set to a starting value of $\eta = 0.01$, which is changed throughout the training milestones to optimize convergence. Last but not the least, cross-entropy loss was used, which is standard for classification tasks. This means that the training accuracy has increased a lot throughout the epochs, reaching around 89% towards the end of training, whereas validation accuracy showed an upward trend and reached a peak at around 88% . This can be interpreted to mean that the model generalizes well to unseen data.

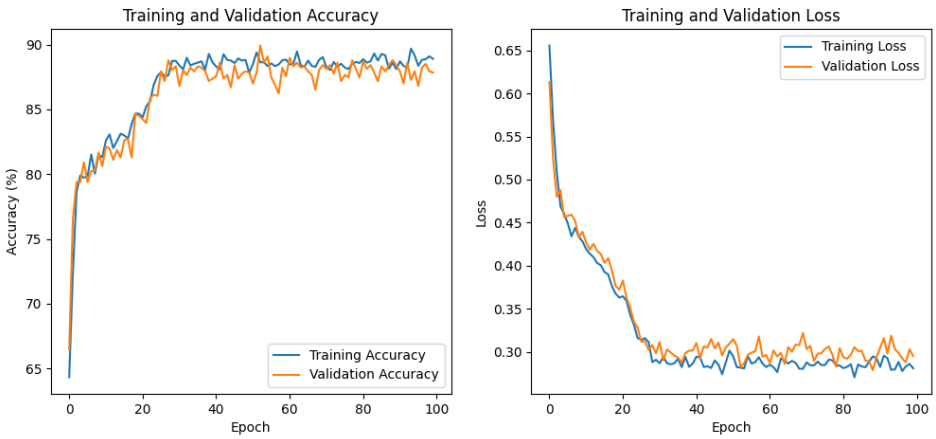

Figure 5.4: Accuracy & Loss Curve of DenseNet121's Training and Validation Data

The training and validation loss plots Figure 5.4 show a consistent decrease in loss with every epoch, which is indicative of effective learning. The training loss decreased from approximately 0.6558 to about 0.2807 and the validation loss seemed to follow a somewhat similar trend, suggesting that the model was, in fact, not overfitting at all despite the complexity of the task.

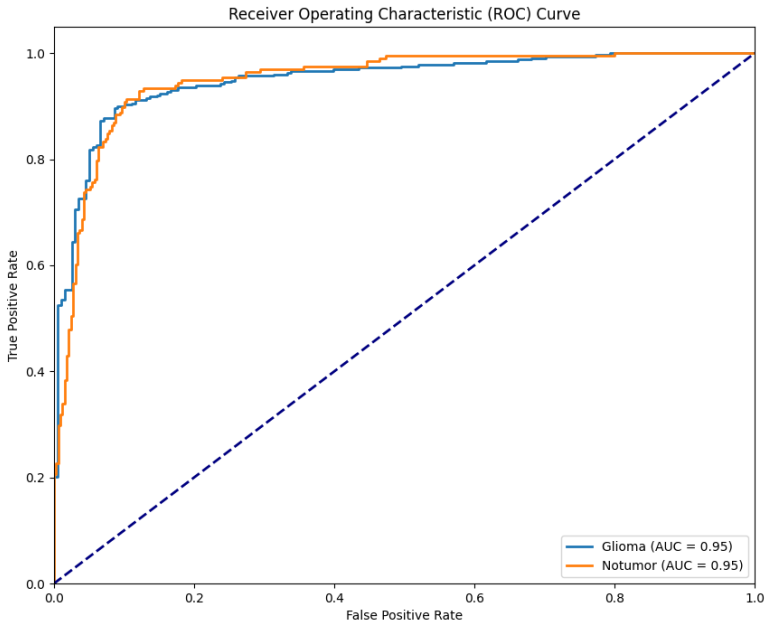

Figure 5.5: ROC Curve of DenseNet121

The ROC curve in Figure 5.5 demonstrates the trade-off between sensitivity and specificity at different threshold settings. The values of AUC (Area Under Curve) close to 1 denote excellent discriminatory ability among classes, further reinforcing the effectiveness of BWN-DenseNet121 in glioblastoma detection.

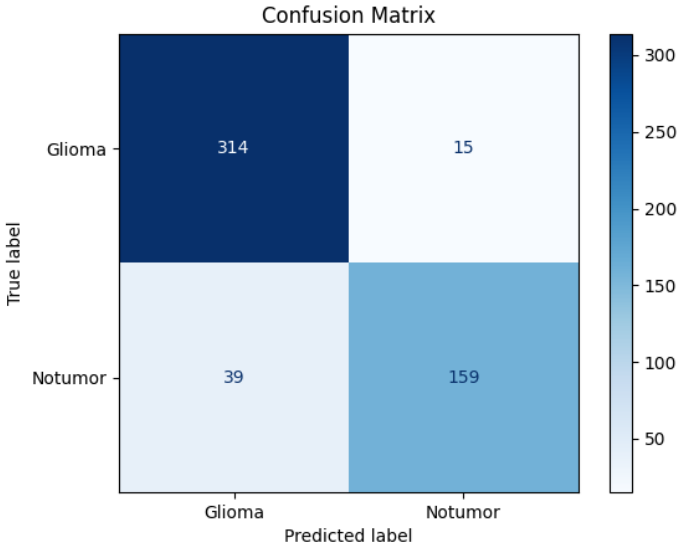

Figure 5.6: Confusion Matrix of DenseNet121

The confusion matrix in Figure gives the distribution of predictions across the two classes. First of all, it is observed that the True Positives (TP) for gliomas are very high, which is an indication of better detection capability. Secondly, the model shows some misclassifications for non-tumor cases that are revealed by overall lower recall for the "No-tumor" class, which indicates room for improvement in terms of reducing false negatives.

The results of glioblastoma detection from MRI images seem promising for the BWN-DenseNet121 architecture with high values of accuracy and precision while keeping the efficiency suitable for mobile applications. Results indicate the possible clinical applications, although further work might be required for improving non-tumor identifications. Some future work may involve addressing the class imbalance to improve the recall for the less represented classes; this could ensure the capability for robust detection across diverse patient populations.

5.1.3 BWN VGG16

We trained our VGG16 model on the dataset and evaluated the model's performance, in terms of several quantitative metrics like accuracy, ROC-AUC curve and Confusion Matrix with the test set.

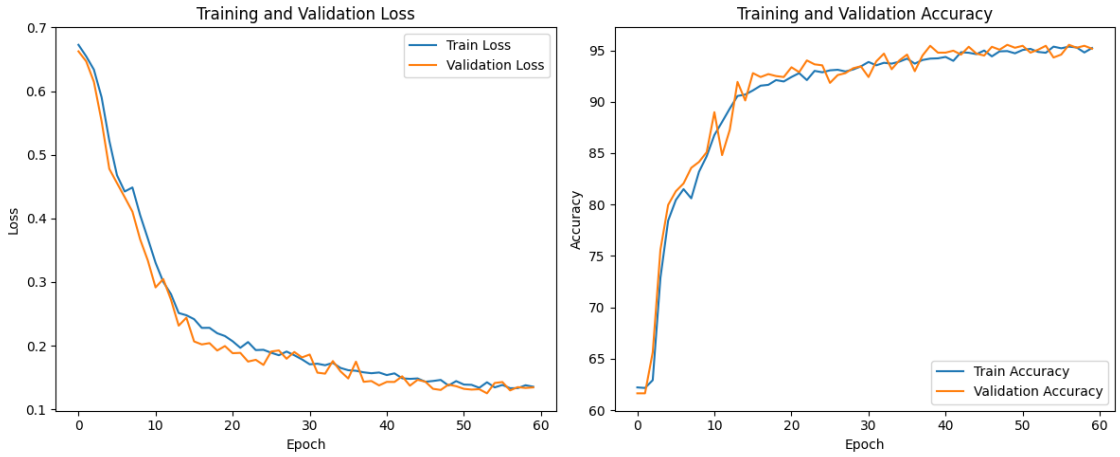

Figure 5.7: Accuracy & Loss Curve of VGG16's Training and Validation Data

To train the model based on the VGG16, we started with an initial learning rate of 0.001 and decreased it gradually throughout a total of 60 epochs (shown in Figure 5.7). On the training data, the model achieved roughly 95.23% accuracy with losses which steadily reduced over every epoch as the learning improved to almost 95.17% accuracy on the validation data. Specifically, we have 0.1356 in the training loss and 0.1345 in the validation loss, which means the model has learned well from data and was able to extract good features for the model.

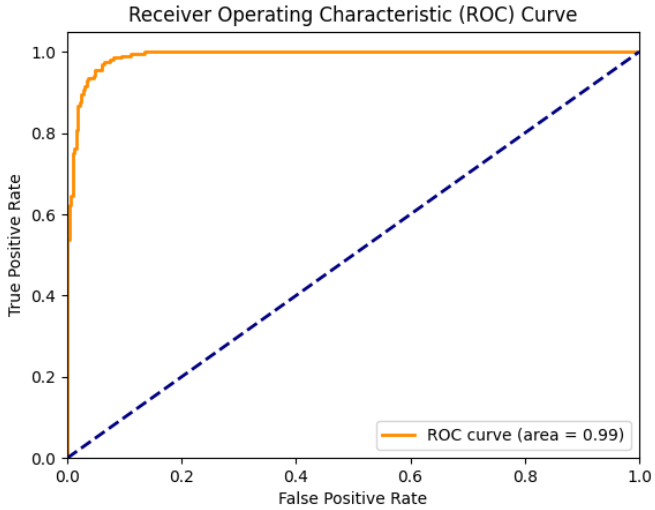

Figure 5.8: ROC Curve of VGG16

Figure 5.8 displays the ROC curve where False Postitve Rate (FPR) is plotted on x-axis and True Positive Rate (TPR) is plotted on y-axis for classification measurement. For the BWN VGG16 model's ability to detect positive Glioma cases. The AUC is 0.99, reaching almost to 1 to the top left corner of the TPR. It predicts that the model has a very high capability of predicting the positive Glioma cases.

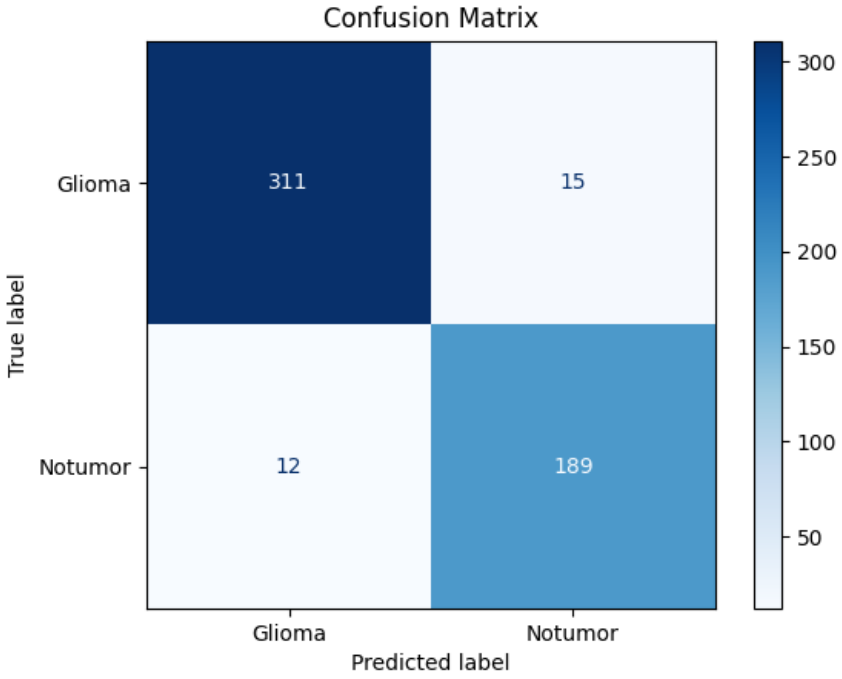

Figure 5.9: Confusion Matrix of VGG16

The high prediction accuracy of the BWN VGG16 model is finally shown in the Figure 5.9. By correctly identifying 311 true positive Glioma cases (326 cases), and 189 true negative No Tumor cases (201 cases) with a small number of misclassifications.

These numbers show how the model has learnt in predicting both classes accurately, while keeping the misclassification quite small.

5.2 Comparison with Full-Precision Models

Model	Binary Model				Full Precision Model			
	Accuracy	Precision	Recall	F1-Score	Accuracy	Precision	Recall	F1-Score
ResNet18	89%	0.899	0.898	0.896	97.59%	0.970	0.983	0.976
DenseNet121	85%	0.853	0.850	0.846	96.81%	0.975	0.99	0.983
VGG16	87%	0.926	0.940	0.933	97.93%	0.98	0.944	0.9634

Table 5.1: Result Comparison of Binary and Full Precision Models

In Table 5.1, We described the classification performance for glioma and no-tumour cases of our binarized models. For BWN-ResNet18, glioma classification, the model achieves a precision of 0.899 ,recall of 0.898 ,and an F1 score of 0.896 . Where it's full precision version these values are 0.973 , 0.983, and 0.976. Regarding BWN-ResNet18, it shows a precision of 0.853, recall of 0.850 ,and an F1 score of 0.846 while the standard version of the model shows a precision of 0.975 , recall of 0.99 and an F1 score of 0.983 . Lastly, BWN VGG16 model provides a precision of 0.926 recall of 0.940 and an F1 score of .0933 where it's full precision version of the model shows a precision of 0.98 , recall of 0.944 ,and an F1 score of 0.964 . Additionally, in terms of accuracy, our proposed binarized models shows a lower value compared to the standard full precision models which are 97.59% for ResNet18, 96.81% for DenseNet121, and 97.93% respectively.

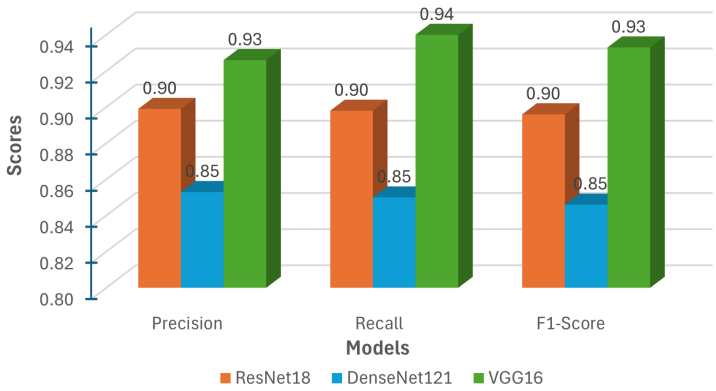

Figure 5.10: Comparison of Precision, Recall, and F1-Score

5.3 Efficiency Analysis

That is, for a standard convolution, the total number of computations equals $c \cdot N_w \cdot N_1$, where the number of channels is c, and filter size (width x height), denoted by $N_w = w \cdot h$; input dimensions (input width x height) are given by $N_1 = w_m \cdot h_m$. This is because, on some modern CPUs, multiplication and addition can execute in a single cycle. However, such efficiency is not applicable to binary-weight networks. Hence, acceleration in Binary Weight Networks (BWN) on those CPUs is not achieved as much.

In our binary convolution approximation, we do $c \cdot N_w \cdot N_1$ binary operations in addition to N_1 non-binary operations. Given that modern CPUs can execute 64 binary operations within a single clock cycle, the achieved speedup can be calculated as:

$$S = \frac{c \cdot N_w \cdot N_1}{\frac{1}{64} \cdot c \cdot N_w \cdot N_1 + N_1} = \frac{64 \cdot c \cdot N_w}{c \cdot N_w + 64} \tag{5.3.1}$$

The key variables that define speedup are channel size (c) and filter size (N_w), while the dimensions of the input are irrelevant. For example, taking standard convolution parameters in the BWN-DenseNet121 model, the channel size c is 256 and the filter size N_w is 9 (3x3 filter), the speedup can be computed as:

$$S = \frac{64 \cdot 256 \cdot 9}{256 \cdot 9 + 64} = \frac{147,456}{2,368} \approx 62.27 \tag{5.3.2}$$

The calculation implies that the approximation might achieve a theoretical speedup of about 62.27 times. However, several overheads usually limit this in actual implementation, like memory access and allocation. Thus, an approximate speed-up of about 58 times may be seen for practical CPU implementations without including memory management processes.

It must be said that when the channel size is reduced (e.g., $c = 3$, commonly found in the initial convolutional layer), the theoretical speedup is substantially reduced. In the case where $c = 3$ and $N_w = 9$, the speedup would be:

$$S = \frac{64 \cdot 3 \cdot 9}{3 \cdot 9 + 64} = \frac{1,728}{91} \approx 18.99 \tag{5.3.3}$$

The reduced speedup for such cases illustrates the reasons for the avoidance of binarization within both the first and last layers of the network: the first layer contains small channel sizes, while the last layer contains small filter sizes, usually 1x1. This approach is very effective in maintaining both efficiency and accuracy.

Furthermore, binary-weight networks demonstrate far better memory efficiency compared to traditional networks relying on double-precision weights. In Figure 5.11, we showed a comparison between different neural networks of memory usage, between usual ones and binarized ones.

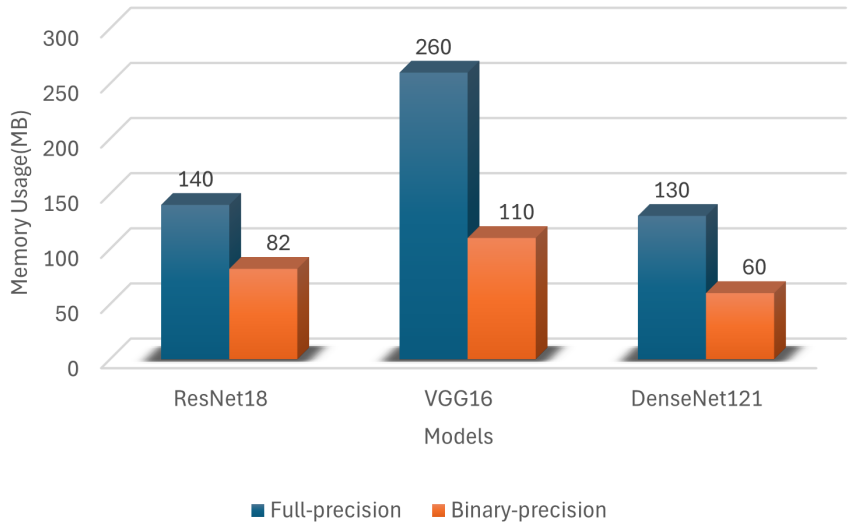

Figure 5.11: Memory Usage Comparison- Full Precision vs. Binary Weight Networks

This demands a great deal of the memory; hence, these networks are best fitted to portable devices with low memory capacity, such as smartphones and embedded systems. Thus, a theoretical analysis favors a heavy computational advantage for binary convolutions, though their practical implementation should optimize hardware-specific constraints and overheads for effective speedup. We used a similar strategy. Binary weighted neural networks are so small that they can be easily fitted into a portable device.

5.4 Layer-wise Binarization Impact

5.4.1 ResNet18

Model	Accuracy(%)	Inference Time (ms)	Memory Usage (MB)
Original Model	98	82	140
Initial Conv Binarized	95	74	130
FC Binarized	92	54	100
All Layers Binarized	89	42	82

Table 5.2: Layer-wise Binarization Impact Analysis for ResNet18

Table 5.2 describes how binarization impacts accuracy, inference time, and memory usage on the model. On the first row, the original model depicts the full precision values for convolution and fully connected layers. When the model is not binarized, and is computed using full precision values, it shows almost 98% accuracy, 82 ms of inference time and 140 MB memory usage to load the weight. For the second row, only the initial convolution layers were binarized while the fully connected layer is computed with full precision values which produces 95% of accuracy, 74 ms of inference time and 130 MB of memory usage per epoch. In contrast to the second

row, initial convolution is kept in full precision making the fully connected layers binarized. This change results in 92% accuracy, 54 ms inference time and a memory usage of 100 MB. It is evident that, with the binarization process, even though there is a slight reduction in accuracy, the inference time and memory usage are decreasing as well, which primarily matches with our goal of building an efficient and lightweight model for resource-constrained systems. Nonetheless, the last row where both the convolution and fully connected are binarized, shows 89% accuracy, 42 ms inference time and 82 MB of memory usage with a notable amount of less inference time and memory usage.

5.4.2 DenseNet121

Model	**Accuracy (%)**	**Inference Time (ms)**	**Memory Usage (MB)**
Original DenseNet121	97	95	150
Conv0 Binarized	94	88	140
Early Blocks Binarized	92	75	120
Late Blocks Binarized	91	73	115
Fully Connected Binarized	93	70	110
All Layers Binarized	85	60	90

Table 5.3: Layer-wise Binarization Impact Analysis for DenseNet121

Table 5.3 describes how binarization impacts accuracy, inference time, and memory usage on the DenseNet121 model. On the first row, the original model depicts the full precision values for convolution and fully connected layers. When the model is not binarized, and is computed using full precision values during forward pass, it shows almost 97% accuracy, 95 ms of inference time and 150 MB memory for loading the weight. For the second row, only the initial convolution layers were binarized while the blocks and fully connected layers are computed with full precision values which produces 94% of accuracy, 88 ms of inference time and 140 MB of memory usage. Now, in the third and fourth row, where early blocks are binarized keeping all other layers in full precision, generates better performance in terms of accuracy, inference and memory usage compared to the one where only later blocks are binarized keeping all other layers in full precision. In contrast to the second row, the fifth row shows the values where initial convolution and blocks are kept in full precision making the fully connected layers binarized. This change results in 93% accuracy, 70 ms inference time and 110 MB usage of memory. It is evident that with the binarization process, even though there is a slight reduction in accuracy, the inference time and memory usage are decreasing as well, which primarily matches with our goal of building an efficient and lightweight model for resource constrained systems. Finally, the last row where both the all layers are binarized, shows 85% accuracy, 60 ms inference time and 90 MB of memory usage for weight loading with a notable amount of less inference time and memory usage.

5.4.3 VGG16

Model	Accuracy (%)	Inference Time (ms)	Memory Usage (MB)
Original VGG16	98	110	260
Conv0 Binarized	96	100	220
Early Layers Binarized	92	85	190
Middle Layers Binarized	94	80	145
Fully Connected Binarized	91	78	115
All Layers Binarized	87	65	110

Table 5.4: Layer-wise Binarization Impact Analysis for VGG16

Table 5.4 shows how binarization impacts accuracy, inference time, and memory usage on the VGG16 model. On the first row, the original model depicts the full precision values for convolution and fully connected layers. When the model is not binarized, and is computed using full precision values during forward pass, it shows almost 98% accuracy, 110 ms of inference time and 260 MB memory usage for loading weights, which requires powerful devices. For the second row, only the initial convolution layers were binarized while the later layers and fully connected layers are computed with full precision values, producing 96% of accuracy, 100 ms of inference time and 220 MB of memory usage. Now, in the third and fourth row, where early layers are binarized keeping all other layers in full precision, generates better performance in terms of accuracy, but a high inference and memory usage value compared to the one where only later layers are binarized keeping all other layers in full precision. Such a high value in inference and memory usage is something that we are trying to solve by binarizing the weights. In contrast to the second row of the initial convolution layer, the fifth row of binarized fully connected layers shows a bit of low accuracy of 91% even though low inference of 78 ms and 115 MB memory usage is reaching toward our goal of using less computation power. values where initial convolution and blocks are kept in full precision making the fully connected layers binarized. Lastly, the row at the end where all the layers are binarized, shows 87% accuracy, a notable amount of less inference time of 65 ms, and 110 MB of memory usage. As VGG16 has dense layer connectivity and uses huge mathematical operations, inference, and memory usage are higher for large computation purposes.

5.5 Qualitative Analysis

5.5.1 Sample Predictions

Figure 5.12 shows some random predictions on our test data made using the ResNet18 model. Each image shows the prediction and actual class to identify successful classifications and misclassifications. In the first image, the model misclassified the image as “Glioma”, whereas the actual class is “No-tumor”. The second image, on the other hand, was correctly classified as a “Glioma”. The third and fourth images are also classified correctly according to their actual class. The first and fourth images demonstrate the model’s ability to distinguish between “Glioma” and “No-tumor”.

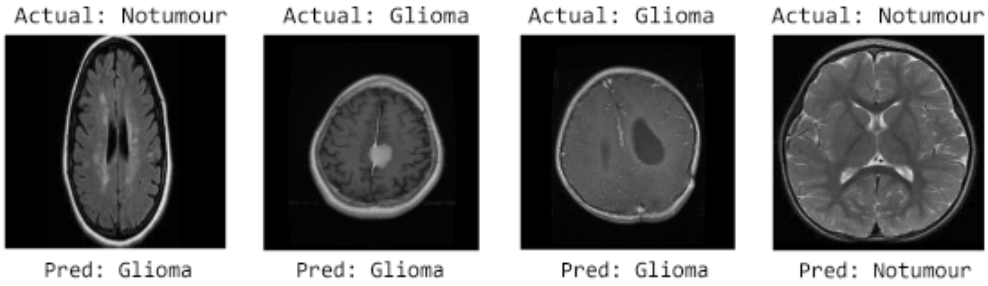

Figure 5.12: Visual Examination of ResNet18 Predictions

Our model mistakenly identified the first image as "No-tumor". However, the second, third, and fourth images were correctly predicted, indicating that the model can accurately classify "Glioma" and "No-tumor" when the image features are clear.

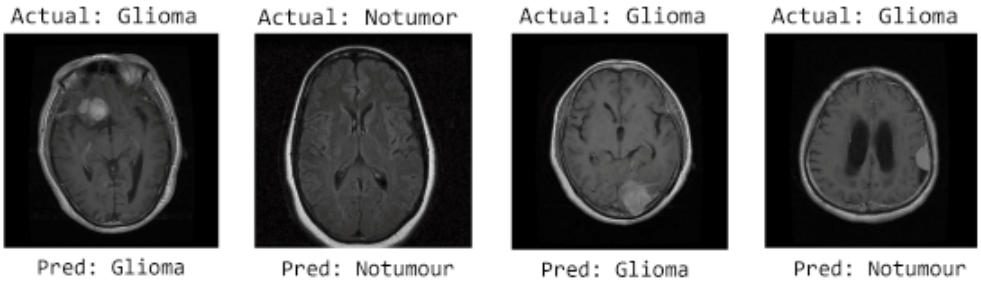

Figure 5.13: Visual Examination of DenseNet121 Predictions

Figure 5.13 shows some random predictions on our test data made using the DenseNet121 model. In the last image, the model misclassified the image as "No-tumor", whereas the actual class is "Glioma". The first, second and third images are classified correctly according to their actual class. The first, third and fourth images demonstrate the model's ability to distinguish between "Glioma" and "No-tumor". Our model mistakenly identified the last image as "No-tumor." However, the second, third, and fourth images were correctly predicted, indicating that the model can classify "Glioma" and "No-tumor" accurately when the image features are clear.

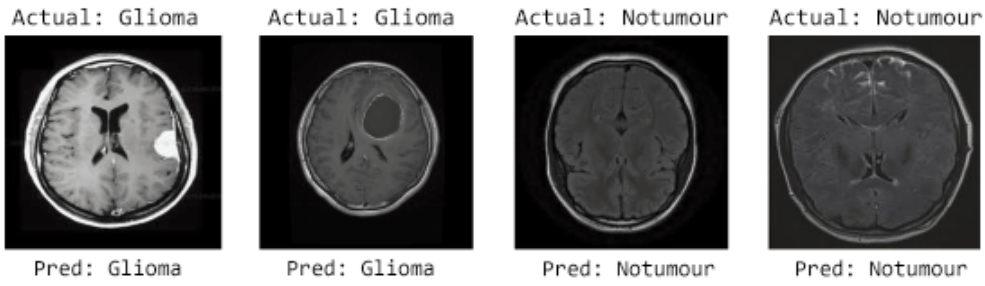

Figure 5.14: Visual Examination of VGG16 Predictions

Figure 5.14 shows some random predictions on our test data made using the VGG16 model. The model correctly predicted "Glioma" for both images, where the actual label is "Glioma" in the first two images. This indicates that the model can diagnose "Glioma" well when the features of tumors are clear and visible. The successful classification results seen in these examples indicate that our model can detect significant patterns linked with glioma and imply that this method may work well on patients. Images one and two are "Glioma" cases (* Both labels: "Glioma"), whereas Images three and four are "No-tumor" cases (**Both labels: "No-tumor"). Second, our model's successful classification of these new samples shows that it is also good at detecting the difference between "Glioma" and "No-tumor" cases.

5.5.2 Error Analysis

We can see that our models sometimes give false positive results and sometimes false negative results. The possible reasons we identified are as follows:

Data Quality Issues: The images that are misclassified may have noise or patterns that resemble glioma, causing the model to misinterpret the image class. This can be mitigated by enhancing data quality and applying more aggressive data preprocessing.

Model Limitations: As our model is binarized, it may lose slight variations and details of the input data, which are important for accurate classification.

Challenging Imaging Characteristics: MRI images can vary greatly due to different imaging conditions and the use of different scanners, which can introduce artifacts and features that confuse the model.

Chapter 6

Future Work and Conclusion

6.1 Limitations

While modern artificial intelligence is finding increased adoption into medical diagnostics, it is bound to go a long way in enhancing the healthcare system. The deployment of these deep learning models at edge devices or embedded systems will do a lot in further advancing real-time service delivery. However, our present work, focusing on the development, training, and validation of the binarized CNN models for Glioblastoma detection, points to various limitations. The first important limitation is an inherent dependency on edge computing environments. While edge computing reduces reliance on cloud infrastructure, it introduces challenges related to model optimization for various hardware specifications.

Additionally, the performance of binarized models might vary based on edge computational capabilities, and diagnostic precision may be at stake. Furthermore, medical imaging data is complex; therefore, good pre-processing and feature extraction techniques must be ensured to warrant reliable results. Another limitation can be large requirements of training datasets, reducing the general applicability of the models across a population and different imaging protocols.

While edge computing seeks to enhance privacy by conducting most of the processing locally, it does not avoid all considerations regarding data security and compliance with healthcare regulations. Having discussed all these, we shall further point out the direction of future work to perfect our binarized models for a host of edge computational environments and make them even more practical and efficient.

6.2 Future Work

As medical diagnostics is adapting modern artificial intelligence for a better health care system, deployment of deep learning models on edge devices or embedded systems add a promising advancement for real-time, easy service reception. While primarily, our current work emphasizes on building, training and validating binarized CNN models for brain tumor (Glioblastoma) detection, future improvements will concentrate on optimizing these models for edge computing environments.

The main benefit of edge computing includes reduced cloud computing dependency. Large models and datasets take so much time to be processed for detection purposes, which are sent to cloud or central servers to do the actual computations. This whole process of sending user data to the central server and receiving the result in the user device causes cloud dependency which involves - latency in real-time diagnosis, problematic access to the central server in remote areas due to limited internet connectivity. Furthermore, such transmission of sensitive medical data raises privacy and security concerns with the possibility of unauthorized access.

For these constraints, edge computing provides an optimal solution. As the diagnostics or computation can be done locally on users device instead of sending data to cloud servers, this shows several advantages, as in - faster response time, less dependency and easy access on cloud servers just to send the outcomes of computed data, reduction of data breachment risk. By implementing edge computing, we hope that our binarized models would be able to detect Glioblastoma efficiently in embedded systems.

6.3 Conclusion

This work, therefore, presents a leap ahead in neuroimaging technology since it establishes a valid route toward early and reliable diagnosis of GBM for improved patient care. Accordingly, this study uses deep learning methodologies in innovative ways to illustrate the possibility of improvement in diagnosis accuracy and efficiency regarding the detection of brain tumors. Unique approaches, specifically tailored to the challenges associated with GBM detection, are being harnessed in this work to address critical gaps in existing diagnostic practices.

Moreover, this study presents an excellent tool and ongoing efforts in continued addressing of the challenges of detection of Glioblastoma, considering the wide-ranging review that has been carried out with a large dataset. These results further suggest that integrating sophisticated computational methodologies into clinical practice may facilitate better patient outcomes in the battle against this insidious type of cancer. Because healthcare systems are now increasingly turning to artificial intelligence for support, these insights provided by this research contribute to a broadened knowledge of how technology can be harnessed to improve both diagnostic processes and strategies for better patient management.

The work, thus, depicts the capability of deep learning in medical imaging and forms a foundation for further steps that may be taken in the time to come. It will definitely provide a way for exploring and refining these models, ultimately aiming at bringing about improvement in early detection through a lightweight model and treatment options available to patients suffering from Glioblastoma.

Bibliography

[1] I. C. London, *Ixi dataset*, Dataset, 2012. [Online]. Available: https://brain-development.org/ixi-dataset/.

[2] M. Courbariaux, Y. Bengio, and J.-P. David, "Binaryconnect: Training deep neural networks with binary weights during propagations," *Advances in neural information processing systems*, vol. 28, 2015.

[3] X. Liang, B. Cui, Y. Lv, and Y. Fu, "Research on the collaborative analysis technology for source code and binary executable based upon the unified defect mode set," in *2015 9th International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*, 2015, pp. 260–264. DOI: 10.1109/IMIS.2015.40.

[4] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio, "Binarized neural networks," vol. 29, D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, Eds., 2016. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2016/file/d8330f857a17c53d217014ee776bfd50-Paper.pdf.

[5] D. N. Louis, A. Perry, G. Reifenberger, *et al.*, "The 2016 world health organization classification of tumors of the central nervous system: A summary," *Acta neuropathologica*, vol. 131, pp. 803–820, 2016.

[6] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi, "Xnor-net: Imagenet classification using binary convolutional neural networks," in *European conference on computer vision*, Springer, 2016, pp. 525–542.

[7] L. Scarpace, T. Mikkelsen, S. Cha, *et al.*, *The cancer genome atlas glioblastoma multiforme collection (tcga-gbm) (version 5) [data set]*, https://doi.org/10.7937/K9/TCIA.2016.RNYFUYE9, Dataset, 2016. DOI: 10.7937/K9/TCIA.2016.RNYFUYE9.

[8] L. Baruah, "Performance comparison of binarized neural network with convolutional neural network," 2017.

[9] J. Shell and W. D. Gregory, "Efficient cancer detection using multiple neural networks," *IEEE Journal of Translational Engineering in Health and Medicine*, vol. 5, pp. 1–7, 2017. DOI: 10.1109/JTEHM.2017.2757471.

[10] W. Tang, G. Hua, and L. Wang, "How to train a compact binary neural network with high accuracy?" In *Proceedings of the AAAI conference on artificial intelligence*, vol. 31, 2017.

[11] J. E. Villanueva-Meyer, M. C. Mabray, and S. Cha, "Current clinical brain tumor imaging," *Neurosurgery*, vol. 81, no. 3, pp. 397–415, 2017.

[12] J. Bethge, M. Bornstein, A. Loy, H. Yang, and C. Meinel, "Training competitive binary neural networks from scratch," *arXiv preprint arXiv:1812.01965*, 2018.

[13] J. Kung, D. Zhang, G. Van der Wal, S. Chai, and S. Mukhopadhyay, "Efficient object detection using embedded binarized neural networks," *Journal of Signal Processing Systems*, vol. 90, pp. 877–890, 2018.

[14] T.-D. Nguyen and E.-N. Huh, "Ecsim++: An inet-based simulation tool for modeling and control in edge cloud computing," in *2018 IEEE International Conference on Edge Computing (EDGE)*, 2018, pp. 80–86. DOI: 10.1109/EDGE.2018.00018.

[15] A. Prabhu, V. Batchu, R. Gajawada, S. A. Munagala, and A. Namboodiri, "Hybrid binary networks: Optimizing for accuracy, efficiency and memory," in *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2018, pp. 821–829. DOI: 10.1109/WACV.2018.00095.

[16] H. Wang, Y. Xu, B. Ni, L. Zhuang, and H. Xu, "Flexible network binarization with layer-wise priority," in *2018 25th IEEE International Conference on Image Processing (ICIP)*, 2018, pp. 2346–2350. DOI: 10.1109/ICIP.2018.8451576.

[17] A. Bulat and G. Tzimiropoulos, "Xnor-net++: Improved binary neural networks," *arXiv preprint arXiv:1909.13863*, 2019.

[18] Y. Li, Z. Liu, W. Liu, *et al.*, "A 34-fps 698-gop/s/w binarized deep neural network-based natural scene text interpretation accelerator for mobile edge computing," *IEEE Transactions on Industrial Electronics*, vol. 66, no. 9, pp. 7407–7416, 2019. DOI: 10.1109/TIE.2018.2875643.

[19] P. Prasanna, A. Karnawat, M. Ismail, A. Madabhushi, and P. Tiwari, "Radiomics-based convolutional neural network for brain tumor segmentation on multiparametric magnetic resonance imaging," *Journal of Medical Imaging*, vol. 6, no. 2, p. 024 005, 2019. DOI: 10.1117/1.JMI.6.2.024005. [Online]. Available: https://doi.org/10.1117/1.JMI.6.2.024005.

[20] T. Simons and D.-J. Lee, "A review of binarized neural networks," *Electronics (Basel)*, vol. 8, no. 6, p. 661, Jun. 2019. DOI: 10.3390/electronics8060661.

[21] J. Bethge, C. Bartz, H. Yang, Y. Chen, and C. Meinel, "Meliusnet: Can binary neural networks achieve mobilenet-level accuracy?" *arXiv preprint arXiv:2001.05936*, 2020.

[22] G. Cerutti, R. Andri, L. Cavigelli, E. Farella, M. Magno, and L. Benini, "Sound event detection with binary neural networks on tightly power-constrained iot devices," in *Proceedings of the ACM/IEEE International Symposium on Low Power Electronics and Design*, 2020, pp. 19–24.

[23] Y. Fu, D. Tian, X. Duan, *et al.*, "A camera–radar fusion method based on edge computing," in *2020 IEEE International Conference on Edge Computing (EDGE)*, 2020, pp. 9–14. DOI: 10.1109/EDGE50951.2020.00009.

[24] Z. Liu, Z. Shen, M. Savvides, and K.-T. Cheng, "Reactnet: Towards precise binary neural network with generalized activation functions," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIV 16*, Springer, 2020, pp. 143–159.

[25] Z. Liu, Z. Shen, M. Savvides, and K.-T. Cheng, "Reactnet: Towards precise binary neural network with generalized activation functions," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIV 16*, Springer, 2020, pp. 143–159.

[26] B. Penkovsky, M. Bocquet, T. Hirtzlin, *et al.*, "In-memory resistive ram implementation of binarized neural networks for medical applications," in *2020 Design, Automation Test in Europe Conference Exhibition (DATE)*, 2020, pp. 690–695. DOI: 10.23919/DATE48585.2020.9116439.

[27] H. Phan, Z. Liu, D. Huynh, M. Savvides, K.-T. Cheng, and Z. Shen, "Binarizing mobilenet via evolution-based searching," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 13 420–13 429.

[28] Y.-F. Qin, H. Bao, F. Wang, J. Chen, Y. Li, and X.-S. Miao, "Recent progress on memristive convolutional neural networks for edge intelligence," *Advanced Intelligent Systems*, vol. 2, no. 11, p. 2 000 114, 2020.

[29] M. Goryawala, B. Roy, R. K. Gupta, and A. A. Maudsley, "T1-weighted and t2-weighted subtraction mr images for glioma visualization and grading," *Journal of Neuroimaging*, vol. 31, no. 1, pp. 124–131, 2021.

[30] F. Hamanaka, T. Kanamori, and K. Kise, "A low cost and portable mini motor car system with a bnn accelerator on fpga," in *2021 IEEE 14th International Symposium on Embedded Multicore/Many-core Systems-on-Chip (MCSoC)*, 2021, pp. 84–91. DOI: 10.1109/MCSoC51149.2021.00020.

[31] Q. T. Ostrom, G. Cioffi, K. Waite, C. Kruchko, and J. S. Barnholtz-Sloan, "Cbtrus statistical report: Primary brain and other central nervous system tumors diagnosed in the united states in 2014–2018," *Neuro-oncology*, vol. 23, no. Supplement_3, pp. iii1–iii105, 2021.

[32] A. S. Peddinti, S. Maloji, and K. Manepalli, "Evolution in diagnosis and detection of brain tumor–review," in *Journal of Physics: Conference Series*, IOP Publishing, vol. 2115, 2021, p. 012 039.

[33] P. Śledzińska, M. G. Bebyn, J. Furtak, J. Kowalewski, and M. A. Lewandowska, "Prognostic and predictive biomarkers in gliomas," *International journal of molecular sciences*, vol. 22, no. 19, p. 10 373, 2021.

[34] J. Vreča, I. Ivanov, G. Papa, and A. Biasizzo, "Detecting network intrusion using binarized neural networks," in *2021 IEEE 7th World Forum on Internet of Things (WF-IoT)*, 2021, pp. 622–627. DOI: 10.1109/WF-IoT51360.2021.9595961.

[35] M. Weller, M. van den Bent, M. Preusser, *et al.*, "Eano guidelines on the diagnosis and treatment of diffuse gliomas of adulthood," *Nature reviews Clinical oncology*, vol. 18, no. 3, pp. 170–186, 2021.

[36] M. Islam, S. A. Noshin, M. Islam, M. Razy, and S. Antara, "An efficient approach for binary classification in brain tumor detection using convolutional neural network," Ph.D. dissertation, Brac University, 2022.

[37] M. Javaid, A. Haleem, R. P. Singh, R. Suman, and S. Rab, "Significance of machine learning in healthcare: Features, pillars and applications," *International Journal of Intelligent Networks*, vol. 3, pp. 58–73, 2022.

[38] A. Khorasani, R. Kafieh, M. Saboori, and M. B. Tavakoli, "Glioma segmentation with DWI weighted images, conventional anatomical images, and post-contrast enhancement magnetic resonance imaging images by U-Net," *Phys. Eng. Sci. Med.*, vol. 45, no. 3, pp. 925–934, 2022. DOI: 10.1007/s13246-022-01164-w.

[39] M. Rasool, N. A. Ismail, W. Boulila, *et al.*, "A hybrid deep learning model for brain tumour classification," *Entropy*, vol. 24, no. 6, p. 799, 2022.

[40] J. Walsh, A. Othmani, M. Jain, and S. Dev, "Using u-net network for efficient brain tumor segmentation in mri images," *Healthcare Analytics*, vol. 2, p. 100 098, 2022.

[41] L. Zhang, X. Yan, and D. Ma, "A binarized neural network approach to accelerate in-vehicle network intrusion detection," *IEEE Access*, vol. 10, pp. 123 505–123 520, 2022. DOI: 10.1109/ACCESS.2022.3208091.

[42] J. Zhao, S. Xu, R. Wang, *et al.*, "Data-adaptive binary neural networks for efficient object detection and recognition," *Pattern Recognit. Lett.*, vol. 153, pp. 239–245, 2022. DOI: 10.1016/j.patrec.2021.12.012.

[43] R. Marcus, "What is glioblastoma?" *JAMA*, vol. 329, no. 14, p. 1232, 2023. DOI: 10.1001/jama.2023.2234.

[44] J. Nalepa, K. Kotowski, B. Machura, *et al.*, "Deep learning automates bidimensional and volumetric tumor burden measurement from mri in pre-and post-operative glioblastoma patients," *Computers in biology and medicine*, vol. 154, p. 106 603, 2023. DOI: 10.1016/j.compbiomed.2023.106603.

[45] C. Yuan and S. S. Agaian, "A comprehensive review of binary neural network," *Artif. Intell. Rev.*, vol. 56, no. 11, pp. 12 949–13 013, 2023. DOI: 10.1007/s10462-023-10464-w.

[46] S. M. Alzahrani and A. M. Qahtani, "Knowledge distillation in transformers with tripartite attention: Multiclass brain tumor detection in highly augmented mris," *Journal of King Saud University-Computer and Information Sciences*, vol. 36, no. 1, p. 101 907, 2024.

[47] M. A.-Z. Khan, N. Innan, A. A. O. Galib, and M. Bennai, "Brain tumor diagnosis using quantum convolutional neural networks," *arXiv preprint arXiv:2401.15804*, 2024.

[48] J. B. P. Matos, E. B. de Lima Filho, I. Bessa, E. Manino, X. Song, and L. C. Cordeiro, "Counterexample guided neural network quantization refinement," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 43, no. 4, pp. 1121–1134, 2024. DOI: 10.1109/TCAD.2023.3335313.

[49] K. N. Rao, O. I. Khalaf, V. Krishnasree, *et al.*, "An efficient brain tumor detection and classification using pre-trained convolutional neural network models," *Heliyon*, vol. 10, no. 17, 2024.