International Journal of Environmental Sciences
ISSN: 2229-7359
Vol. 11 No. 2, 2025
https://www.theaspd.com/ijes.php

Application of GPU Embedded Systems for Medical Image Analysis: Brain Tumour Segmentation

[1]Dr.Gajula Lakshminarayana,[2] Dr. Praveen Kumar Patidar,[3]PALPANDI S, [4]Mr.M.J.D Ebinezer,[5]Dr.B.YUVARAJ,[6]Dr.R.Senthamil Selvan
[1]Professor,Department of Electronics&Communicaton Engineering,SVR Engineering College,Nandyal,Andhra Pradesh
[2]Associate Professor , Department of Computer Science and Engineering,Parul Institute of Technology, Parul University,Vadodara, Gujarat
[3]Assistant professor-senior Grade, Department of Computer science and Engineering
Vel Tech Rangarajan Dr Sagunthala R&D Institute of Science and Technology, Avadi.
[4]Assistant Professor,Department of Computer Science &amp; Engineering,Koneru Lakshmaiah Education Foundation,Green Fields, Vaddeswaram, A.P
[5]Professor , Department of Information Technology ,Kings Engineering College , Sriperumbudur, Chennai
[6]Dr.R.Senthamil Selvan,Associate Professor,Department of ECE,Annamacharya Institute of Technology and Sciences,Tirupati,Andhra Pradesh
Email:glnarayana.ece@svrec.ac.in,pravinkpatidar@gmail.com,sppecerdeee@gmail.com, ebinezer.mjd@kluniversity.in,byuvarajb@gmail.com,selvasenthamil2614@gmail.com

ABSTRACT:
GPU-embedded devices for image processing in medicine, this work explores accurate brain tumour segmentation, potentially revolutionising neurology's diagnostic precision and treatment planning. This work intends to improve brain tumour segmentation accuracy and efficiency by using GPU processing capability, opening new avenues for medical imaging technology developments. As medical data grows for research and diagnosis, healthcare practitioners automate techniques for reliable and rapid picture analysis, such as segmentation or restoration. Many current solutions for these jobs rely on Deep Learning techniques, which demand strong hardware and are not suitable for the power consumption issues outlined above. Demand exists for developing cost-effective image analysis systems with enhanced performance. The study proposes an automated brain tumour segmentation approach using a Convolutional Neural Network on a low-cost GPU-integrated platform using Deep Learning. By redefining core operations, successfully verified the strategy of segmenting brain tumours using the BRaTS 2022 dataset, demonstrating that artificial neural networks can be trained and deployed in medical fields with minimal resources.

KEYWORDS: GPU, CNN, Deep Learning, ANN, Healthcare practitioner.

1. INTRODUCTION:

Brain tumour segmentation technologies using Deep Learning (DL) algorithms have advanced dramatically in recent years. Some models, such as the Input Cascade CNN and U-Net, demonstrate the promise of Convolutional Neural Networks (CNNs) for detecting common tumours that are hard to segregate owing to their diverse forms and locations. Since model designs have become more complex over time, powerful platforms are needed to execute them, resulting in higher expenses due to increased power usage. To achieve production-level potentials and accessibility, DL-based medical image analysis algorithms must be trainable and executable on restricted computing platforms. Healthcare providers automated image analysis technologies to improve diagnosis speed and accuracy due to the growing amount of medical data. DL-based computer vision systems must be affordable, energy-efficient, and portable to be regarded as legitimate and relevant. The Low-Power Image Recognition Challenge (LPIRC) highlights the significance of developing systems while balancing performance and low power consumption. Technology firms have created and marketed low-power hardware accelerators for deep learning. JAX is a machine learning accelerator that simplifies DL application creation and deployment.

This architecture supports transportation, service, manufacturing, smart city, medical image analysis, and other DL workloads. The TensoRT, CUDA, and cuDNN software libraries accelerate inference. This article initially discusses JAX's advantages for DL application development and execution. Then, to optimise neural networks for deployment on devices with limited resources, examine how to apply model compression approaches and a normalisation strategy. The conclusion provides a compressed neural network architecture training scheme for the development of an end-to-end automated tool for the segmentation of glioma tumours in the brain.

2. METHODOLOGY:

Internet of Things (IoT) gadgets for healthcare, autonomous vehicles, and drones were born out of embedded computer systems, which allowed for greater mobility and power efficiency. As a result of the computational complexity of most standard algorithms, there is still an ongoing problem in developing specialised machine learning applications that can be executed on these types of systems. To design DL applications that are consistent with the memory limits that have been set, it is necessary to reconsider and change some procedures to decrease the number of parameters they need. The effectiveness of the JAX and adjustments to neural network operations for the creation of inconsequential CNNs are examined in this part.

2.1 XAVIER JETSON-AGX

One of the most well-liked affordable machine learning accelerators on the market today is the JAX, because of its distinctive characteristics. Its inconsequential and capacity-efficient CPU-GPU merger supplies better power than FPGAs, which is individual of the key reasons to utilise it as a stage for establishing DL uses. When it comes to preparation and concluding the Deep Learning model's real-time requests, GPU-located embedded estimating tools like the JAX may be the way to go. Firstly, the principle's inconsequential answer is the requirements for deploying miniature or winged ploys, which is mainly because the power wanted to fly a drone increases as allure mass increases. The second excuse is the slightest capacity usage, which goals the decline of energy expenses while lowering thermal issues had connection with extents to a degree autonomous forceful. The JAX is suitable for use in entrenched application growth by way of its inconsequential design and reduced capacity consumption GPU-located entrenched calculating devices like the JAX grant permission be superior real-time Deep Learning model preparation and conclusion. Heavy drones need more power to hover, therefore the system's low weight suits small or flying gadget deployment. The majority of CNN layers including pooling, convolution, deconvolution, normalisation, and activation layers are supported by this architecture. As a result of these features, the JAX is an excellent platform for creating CNN-based image analysis applications. Additionally, the equipment is prepared to accept an NVMe SSD drive, which will expand its memory and enable local data storage and processing a feature that is absent in many real-time DL applications that rely on distant servers to do the same thing as shown in Figure 1. One benefit of collecting and processing data locally is that it eliminates the need to depend on a server response, which may lead to unpredictable latency, connection drops, and potential security issues while data is being requested or written. The main research goals are to develop convolutional neural networks with a smaller memory footprint and to speed up their inference for image segmentation applications. A high-performance deep learning inference runtime called Tensor RT is part of the Jet Pack SDK, which the JAX utilises to achieve the aims. It is recommended that current neural network designs be reevaluated in light of JAX's benefits as a CNN training and inference platform. This may be achieved by reducing component sizes and discovering new regularisation strategies to compensate for missing parameters, resulting in more power-efficient models.

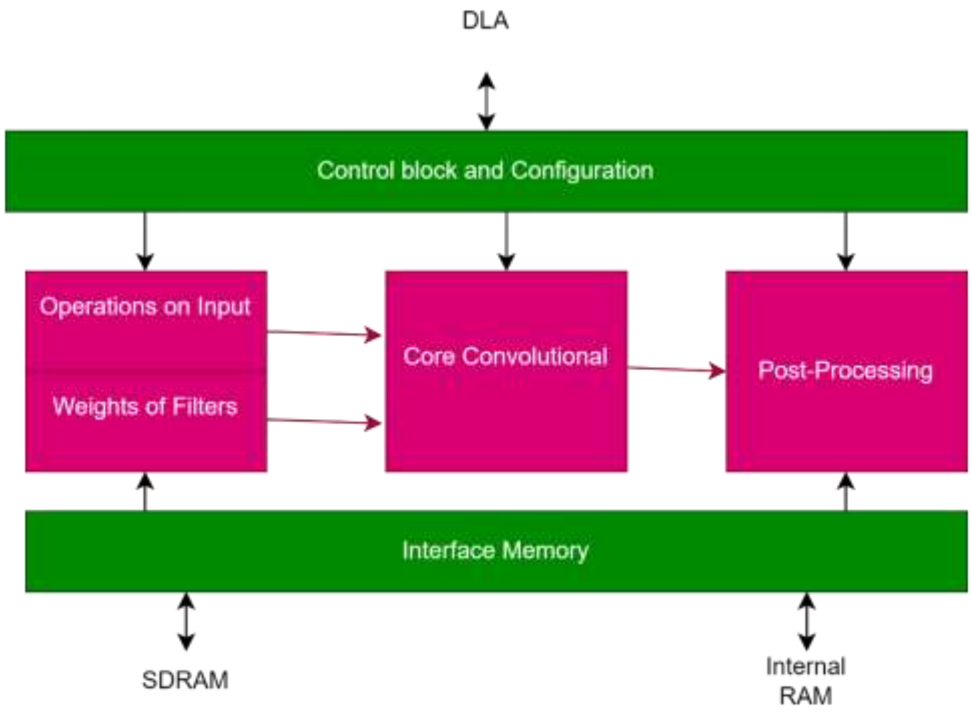

Figure 1: The architecture of the deep learning accelerator from Nvidia

2.2 THE NORMALISATION:

The ability of normalisation approaches to speed up training has made them popular in deep learning. Batch Normalisation (BN) is the most used normalisation technique for training DNNs. By globally normalising features along the batch dimension, BN has shown to be an effective normalisation technique that facilitates the optimisation and convergence of neural networks. Batch statistics' inherent stochasticity makes BN a potent regularizer and aids DNNs in avoiding overfitting. Small mini-batches cause BN to provide erroneous batch statistics estimates and a larger model error, hence big batch sizes are necessary for BN to normalise along the batch dimension. These advantages are listed before are present in this method. Because embedded system training uses a limited amount of memory, it is essential to normalise features along an extra dimension using small input mini-batches without affecting training outcomes adversely. Several approaches have been suggested to address the issue of batch statistics estimation utilising BN with tiny mini-batches, such as Batch Renormalization and Synchronised Batch Normalisation, which aim to achieve improved performance. The size issue can only be redirected or solved by increasing computing power concerning the mini-batch size, but these solutions remain batch-dependent. One normalisation approach for dealing with systems with limited resources is group normalisation (GN). The basic idea behind a GN layer is to group input channels according to their attributes and then normalise them. By avoiding computing the batch statistics, it maintains full independence from the batch dimension. Where Si is the set of pixels used for normalisation and m is the dimensions of the set being managed, the standard deviation σ and the mean μ may be expressed in the same way as in Eq. (1) when a regular feature normalisation is taken into consideration. The characteristics that are being calculated are represented by x, and ε is a tiny constant.

$$\mu_i = \frac{1}{m}\sum_{K\in S_i} x_k, \sigma_t = \sqrt{1/_m}\sum_{K\in S_i}(X_K - \mu_i) + \varepsilon \qquad \text{Eq (1)}$$

Hence, GN is definite as the calculation of σ and μ in the set that is then defined as follows:

$$S_i = k|k_n = i_n, \left|\frac{k_c}{\frac{c}{g}}\right| = | \frac{I_c}{\frac{c}{g}} \qquad \text{Eq (2)}$$

When the batch axis is denoted by N, the group count is denoted by G, and the channel count is equal to G. Be aware that in Eq. (2), a Layer normalisation is used if G ¼ 1, and an Instance normalisation is used if G ½ C, where C is the number of input channels. This approach has shown promise in replacing BN layers in ResNet-50 and other models to achieve reduced training error levels. Since GN serves as both a normalisation and regularisation layer, it speeds up training, eliminates memory limits, and enables DNNs to be trained in smaller groups. To design a regularisation strategy by minimising the association between sets of neurons during training, coupled GN with Dropout to generate Independent-Component (IC) layers. These layers are shown in Figure 2.

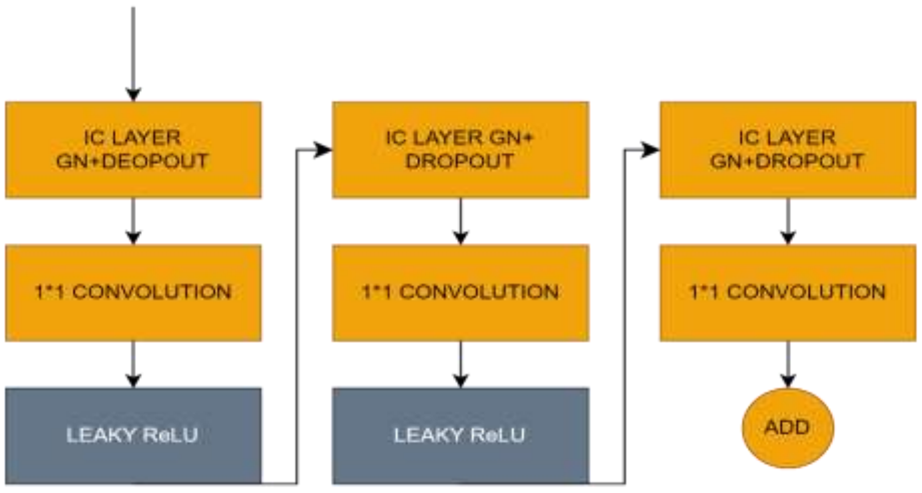

Figure 2: Modified separable difficulty block replaces both conventional and transposed U-net convolutions.

2.3 MODEL COMPRESSOR

Even though convolutional neural networks are effective at performing visual tasks like object identification and picture separation, they still have a hard time running on devices with little processing power. Given that the introduction of deeper models necessitates a rethinking of CNNs for training on these devices and their usage in real-time inference applications. Using the preexisting U-net architecture as a starting point, sought to make it JAX platform compatible for training and inference. Redefining the convolutional procedures using depth-wise separable convolutions was the first approach to compression. Figure 3 shows how this particular sort of convolution may break down normal convolutions into a depth-wise convolution, a regular convolution with a 1 × 1 kernel termed pointwise convolution, and finally, a factorised standard convolution. A depth-wise separable convolution differs from a standard convolution in that it applies a separate filter to each input channel. Equations (3) and (4) for normal and depth-wise separable convolutions establish the total amount of trainable parameters: K, the convolution kernel size, M, the input channels, G, the output image size, and N, the output channels.

.

$$k^2.m.g^2.n \qquad \text{Eq (3)}$$

$$g^2 \cdot m.(k^2 + n) \qquad \text{Eq (4)}$$

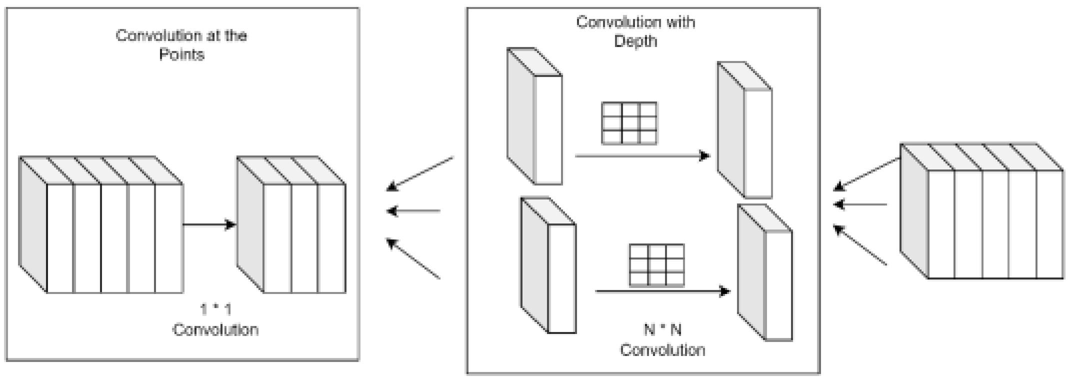

Figure 3: A detachable convolution block is constructed.

The most often used models that use separable convolutions to obtain competitive outcomes in common visual tasks, like ImageNet, are MobileNet and ShuffleNet. These networks take advantage of the compression capabilities. These larger convolution blocks are made up of three smaller ones. After the first 1 × 1 convolution subblock, there are two further types of convolution blocks: one that is depthwise and one that is pointwise. When using a LeakyReLU function, only the regular and depth-wise convolution sub-blocks will be disabled. The IC layer, which was previously discussed, is used by each of the sub-blocks for regularisation and normalisation. Bilinear interpolation was utilised to substitute each transcribed inversion in the decoder element with an upward sampling layer and enhance the bottleneck output maps to minimise network size. Using the procedures that were previously mentioned, Figure 4 illustrates the whole network architecture that is being suggested. Quantization of neural networks was used for inference in addition to compressing the convolution layers. They used an 8-bit quantization after training to further compress the model, which sped up the technique's prediction phase and reduced segmentation time. The reason is, inference performance is unaffected when DNN operations are reduced to an 8-bit fixed point. The use of quantization as a compression approach for neural networks has two benefits: first, it speeds up inference; second, it reduces memory access costs and bandwidth, which helps with power efficiency, which is an issue with most DL applications.

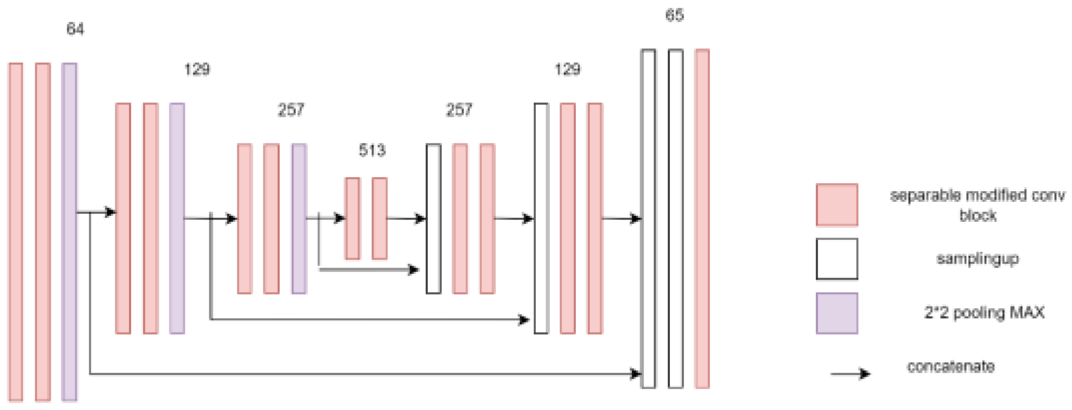

Figure 4: U-network compressed. For decoding, the network utilises bilinear sampling up, max-pooling for downsampling, and MobileNetV2-like separable convolution blocks.

RESULTS AND RESEARCH

2.4 DATA SET

The data collection used for the investigations is the BraTS 2015 dataset. A total of 220 instances of high-grade glioma and 54 instances of low-grade glioma (LGG) make up the training set. Four different sequences, Ability, T2, T1 and T1c are at the disposal. TA ground truth picture with five labels representing tumour structures no tumour (label 0), necrosis (label 1), edoema (label 2), non-enhancing tumour (label 3), and enhancing tumours (label 4) accompanies each brain in the collection. A full tumour area, which includes all four labels, a core tumour region, and an augmenting tumour region are the three categories into which these formations fall. The four sequences and their corresponding pounded truth from an LGG case are shown in Figure 5, with the top row representing the LGG case and the bottom two rows representing the HGG cases. Data was pre-processed before being fed into the neural network. The T1 and T1 c arrangements were corrected using the N4ITK bias field using the Simple ITK package. To avoid overfitting, use data augmentations such as Gaussian noise, unstable to 40% of the slices, elastic transformation, and flipping at random. Since the embedded device must handle data processing locally, a lightweight and quick pipeline for feeding the neural network must be created. In each instance, chose to extract slices by excluding empty slices and cropping them all to 128 × 128 so that wouldn't have to compute the MRI background areas again. As a result, the pipeline generates two datasets—one for training and one for validation featuring HGG and LGG instances in a split ratio of 0.8:0.2. Lastly, the recovered slices are encoded and stored in a Protocol Buffer format to decrease the dataset extent in recollection and improve reading performance for further dispensation.

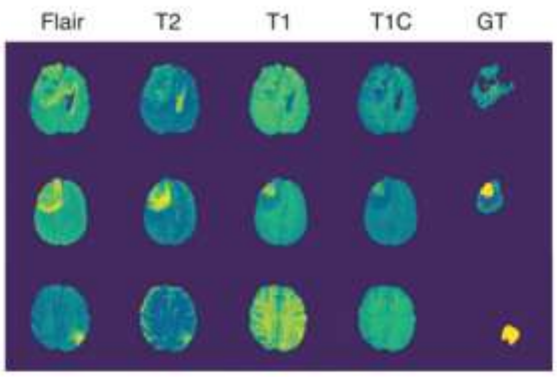

Figure 5: Four

arrangements with pounded truth from the BRaTS 2015 dataset representing LGG and HGG instances.

The model was qualified using an Adam Optimizer with a learning rate of 0.0002. The kernels were initialised using the wonderful normal initialization with all biases set to zero. By combining the Binary Cross-Entropy from Eq. (5) and the Dice loss purposes from Eq. (6), will be able to get higher performance in terms of pixel-wise precision. This loss function is provided by Eq. (7).

$$BCE(y,y) = -\frac{1}{N}\sum_{i+1}^{n}[y_i \log(y_i) + (1-y_i)\log(1-y_i)] \quad \text{Eq(5)}$$

$$dicelos(Y,Y) = 1 - \frac{2\sum Y.Y}{\sum Y+\hat{y}} \quad \text{Eq (6)}$$

$$combinedloss(y,\hat{y} = BCE(y.y)d\imath\widehat{celoss}(y,\widehat{y}) \quad \text{Eq (7)}$$

y stands for the pounded truth labels and ^y for the model predictions, where N is the dataset's sample size. Given that addressed Batch Normalization's shortcomings, permit training the model on 2-item mini-batches. To avoid overfitting, early pausing is used during training to keep an eye on the validation loss values. The model's last layer produces a probabilistic mask pixel-wise, where the values of the individual pixels are determined by a sigmoid activation function. After threshing, a forecast mask containing all five sparse stickers is obtained. Lastly, set the integrated platform to MAXN mode during training, which enables us to utilise each CPU main and raise the GPU frequency, as seen in Figure 6. This allowed it to exploit its potential. The JAX's in-built DL and graphic accelerators, which come with four distinct power envelope configurations, are responsible for this. The model took about 20 minutes for each epoch to train due to this setup. The model's training and inference processes ran well in every mode that was evaluated.

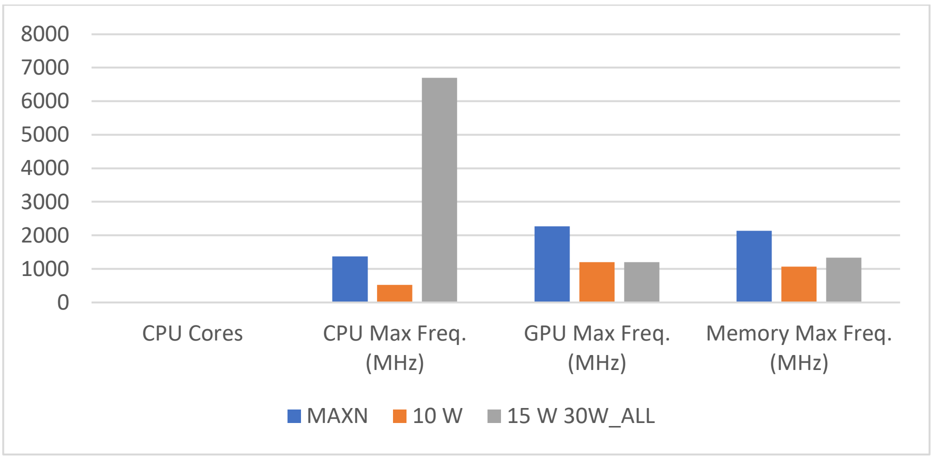

Figure 6: Options for powering up the Nvidia Jetson AGX Xavier

Training and segmentation of high- and low-grade gliomas were both accomplished using the segmentation technique, which made use of the full potential of Janus. Even though the uncompressed

U-Net model has over 310 lakh parameters, the compression method brings that number down to around 210 lakhs, all without significantly impacting the model's inference and training capabilities concerning the restriction reduction factor. This approach is well-suited for use as an embedded device or a portable app for medical image analysis due to its decreased model memory footprint. For each of the previously stated tumour locations, the Dice coefficient was computed using the authentication dataset to demonstrate agreement between the actual and predicted masks. Here are the outcomes:

$$DiceCOef = \frac{2|y \cap \hat{y}}{|y| + |\hat{y}}$$

Figure 7 shows that other segmentation algorithms do better on dice testing, but the method still has a competitive number of parameters needed to accomplish the segmentation job. The compressed model's segmentation capabilities are shown in these three instances: In Figure 8, the input picture is shown in column A as a cropped slice including all four sequences, the ground truth is shown in column B, and the neural network's threshold prediction mask is shown in column C Knowledge Distillation and Parameter Pruning are two further compression strategies that may have been used to further decrease the model footprint. Building and training a DL-based segmentation tool that doesn't rely on any pre-existing models is the main objective of this article. These methods depend on heavy pre-trained networks, which may not be relevant to the challenge, Therefore, it is not advisable to use them for the development of fast real-time applications for devices with limited resources.

.

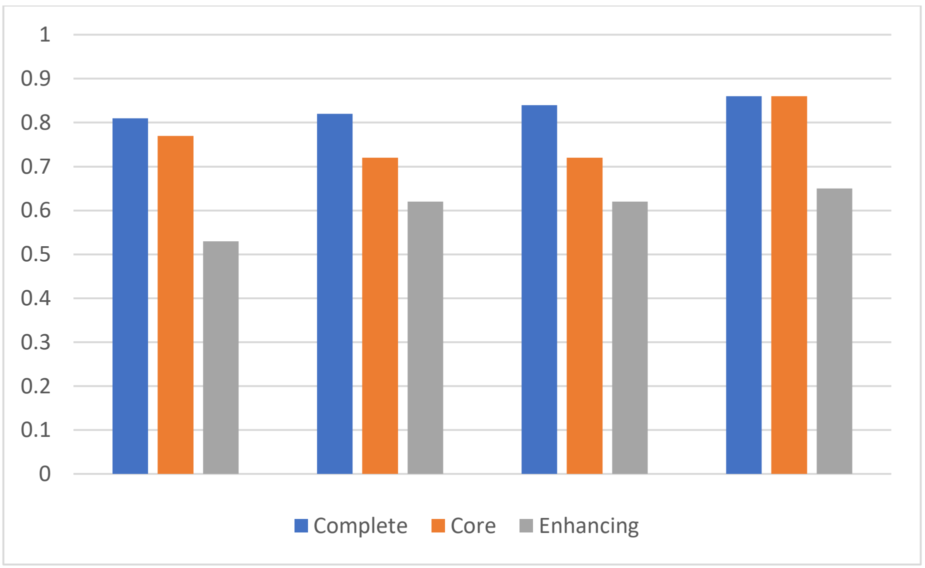

Figure 7: The technique for brain tumour segmentation was compared to existing deep learning-based approaches.

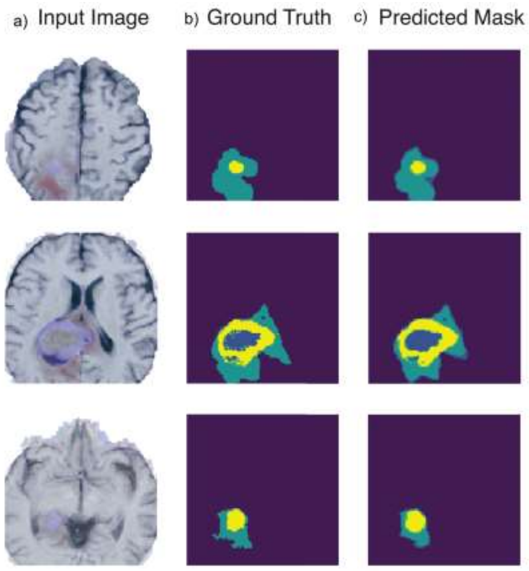

Figure 8: An example of tumour segmentation. Using a 128x128x4 input slice in A, ground truth in B, and compressed model output in C.

3. CONCLUSION:

This study demonstrated that approaches for optimising and compressing neural networks may be effectively used to transfer medical image processing to embedded systems with low resources. To provide a lightweight method for training neural networks, combined an existing CNN design with model compression. A combination of depth-wise separable convolutions and Independent-Component layers may reduce the number of trainable parameters while maintaining good training and inference performance.

After comparing the findings to those of other state-of-the-art methods that use Deep Neural Network training, demonstrated that the technique can achieve comparable results when it comes to the reduction of parameters. The effectiveness of an affordable embedded platform for developing end-to-end DL applications for medical image examination was shown by running a brain tumour segmentation algorithm on the device. This would lead to a decrease in production costs as a result of reduced training and inference time as well as overall energy consumption during the segmentation task.

REFERENCE:

1. Niepceron, Brad, Ahmed Nait-Sidi-Moh, and Filippo Grassia. "Moving medical image analysis to GPU embedded systems: Application to brain tumour segmentation." *Applied Artificial Intelligence* 34.12 (2020): 866-879.
2. Cates, Joshua E., Aaron E. Lefohn, and Ross T. Whitaker. "GIST: an interactive, GPU-based level set segmentation tool for 3D medical images." *Medical image analysis* 8.3 (2004): 217-231.
3. Debnath, Sushanta, and Fazal A. Talukdar. "Brain tumour segmentation using the memory-based learning method." *Multimedia Tools and Applications* 78.16 (2019): 23689-23706.
4. Cabral, Felipe, et al. "Optimized execution of morphological reconstruction in large medical images on embedded devices." *Journal of Real-Time Image Processing* 18.3 (2021): 779-791.
5. Hamdaoui, Fayçal, and Anis Sakly. "Automatic diagnostic system for segmentation of 3d/2d brain MRI images based on a hardware architecture." *Microprocessors and Microsystems* 98 (2023): 104814.
6. Arora, Anuja, et al. "Brain tumour segmentation of MRI images using processed image driven u-net architecture." *Computers* 10.11 (2021): 139.

7. Ali, Owais, et al. "Implementation of a modified U-Net for medical image segmentation on edge devices." *IEEE Transactions on Circuits and Systems II: Express Briefs* 69.11 (2022): 4593-4597.
8. Shi, Lin, et al. "A survey of GPU-based medical image computing techniques." *Quantitative imaging in medicine and surgery* 2.3 (2012): 188.
9. Biswas, Biswajit, Swarup Kr Ghosh, and Anupam Ghosh. "A novel CT image segmentation algorithm using PCNN and Sobolev gradient methods in GPU frameworks." *Pattern Analysis and Applications* 23.2 (2020): 837-854.
10. Radha, R., and R. Gopalakrishnan. "A medical analytical system using intelligent fuzzy level set brain image segmentation based on improved quantum particle swarm optimization." *Microprocessors and Microsystems* 79 (2020): 103283.
11. Kofler, Florian, et al. "Brats toolkit: translating brats brain tumour segmentation algorithms into clinical and scientific practice." *Frontiers in Neuroscience* 14 (2020): 501835.
12. Aboian, Mariam, et al. "Clinical implementation of artificial intelligence in neuroradiology with development of a novel workflow-efficient picture archiving and communication system-based automated brain tumour segmentation and radiomic feature extraction." *Frontiers in Neuroscience* 16 (2022): 860208.
13. Mohan, Geethu, and M. Monica Subashini. "Medical imaging with intelligent systems: a review." *Deep learning and parallel computing environment for bioengineering systems* (2019): 53-73.
14. Saxena, Sanjay, and Sudip Paul, eds. "High-Performance Medical Image Processing." (2022).
15. Nalepa, Jakub, et al. "Fully-automated deep learning-powered system for DCE-MRI analysis of brain tumours." *Artificial intelligence in medicine* 102 (2020): 101769.
16. Menze, Bjoern H., et al. "The multimodal brain tumour image segmentation benchmark (BRATS)." *IEEE transactions on medical imaging* 34.10 (2014): 1993-2024.
17. Niepceron, Brad. *Development of a diagnosis application based on artificial neural networks for the detection of brain tumours*. Diss. Université de Picardie Jules Verne, 2021.
18. Rodríguez Corral, José María, et al. "Energy efficiency in Edge TPU vs. embedded GPU for computer-aided medical imaging segmentation and classification." (2023).
19. Zeineldin, Ramy A., et al. "DeepSeg: deep neural network framework for automatic brain tumour segmentation using magnetic resonance FLAIR images." *International Journal of computer assisted radiology and surgery* 15.6 (2020): 909-920.
20. Guan, Xi, et al. "3D AGSE-VNet: an automatic brain tumour MRI data segmentation framework." *BMC Medical Imaging* 22 (2022): 1-18.
21. R.Senthamil Selvan “Machine Learning-Based Lifespan Prediction Modelling for Electric Submersible Pumps in Oil Wells” by 2024 Second International Conference Computational and Characterization Techniques in Engineering & Sciences (IC3TES), 15-16 November 2024,ISSN:0018-9219,E-ISSN:1558-2256, 17 February 2025, DOI: 10.1109/IC3TES62412.2024.10877544