Faculty Of Computing Informatics

TDS2101 PROJECT REPORT (40%)

Trimester 1, 2022/2023

Project Title: Brain Tumour Detection with Machine Learning

Lecture: Syuhaidah Azni

By: Ayat Abdulaziz Gaber Al-Khulaqi

ID: 1191202335

Abstract

Brain tumor detection using convolutional neural networks and transfer learning. Training and testing the models with Adam and RMSprop optimizers, a number of 64 and 128 batches, and for transfer learning three different architecture models were used. The three architecture models are MobileNetV2, VGG19, and ResNet101V2. Transfer Learning had way better accuracy and loss results than conventional neural networks. However, in transfer learning, ResNet101V2 had the highest accuracy rate at 81.73% with batch number 128 using RMSprop optimizer and the lowest loss rate at 248.79% with batch number 128 using Adam optimizers.

Keywords: Adam, RMSprop, MobileNetV2, VGG19, ResNet101V2, Batch

Table of Contents

1. Introduction

Mass or growth of abnormal cells in your brain are known as a brain tumour. The average size of the brain tumour doubles its size every twenty-five days, due to that the survival rate of the patient is usually not more than a half year if not treated properly [1]. Brain tumours can be malignant which is cancerous or benign which is noncancerous. Malignant brain tumours are considered one of the most deadly cancers that lead to over 14 thousand deaths each year [3].

Magnetic resonance imaging (MRI) is a medical imaging technique that uses a magnetic field and computer-generated radio waves to generate detailed images of the organs and tissues in the body [5]. MRI is handled and examined by doctors, because of the manual detection of doctors, they can not notice the correct position of the growth set within the brain [2]. This leads to the possibility of human error and it raises the overall risk factor of a clinical case, which can sometimes result in tragic possible outcomes. Developing countries have poorly developed health system that includes cancer services, in need of improvement [4]. That increases the possibility of human error. Using deep learning algorithms will help clinical experts to detect the initial stages of the tumour.

For this research project a dataset from the Kaggle website will be used, the dataset includes three different types of tumours which are glioma, meningioma, and pituitary tumours also an MRI of no tumour. Glioma tumour is a malignant (cancerous) type of tumour that happens in the brain and spinal cord. Meningioma tumour also occurs in the brain and spinal cord however in the majority of cases meningioma tumour is benign (noncancerous). Pituitary tumours happen at the pituitary gland near the brain in most cases they are benign (noncancerous) same as meningioma tumours. More about the dataset will be described in the section below this.

Project Objectives

There are three main objectives of this research project which are:

1) To compare and find the higher accuracy rates between the deep learning algorithms.

2) To compare and find the lower loss rates between the deep learning algorithms.

2. Dataset Description

The MRI dataset was acquired from the Kaggle website which will be used for this research project. The dataset contains two directories which are:

Testing Directory Includes:

- Glioma tumour directory contains 100 magnetic resonance images.
- Meningioma tumour directory contains 115 magnetic resonance images.
- Pituitary tumour directory contains 105 magnetic resonance images.
- No tumour directory contains 74 magnetic resonance images.

Training Directory Includes:

- Glioma tumour directory contains 826 magnetic resonance images.
- Meningioma tumour directory contains 822 magnetic resonance images.
- Pituitary tumour directory contains 395 magnetic resonance images.
- No tumour directory contains 827 magnetic resonance images.

3. Research Methodology & Model

Under this section, shows the accuracy and the loss rates using the Adam optimizer, 50 epochs, image size of 224 by 224, and 128 as the batch number. The accuracy and loss rates are displayed using a line chart.

3.1. Conventional neural networks (CNN)

Conventional neural networks (CNN) is a supervised type of Deep learning, mainly used for image and speech recognition.

For the training accuracy:

It starts around 43%, then sharply increases from epoch 0 to 8. From epoch 8 to epoch 13, it slowly increases, then decreases from epoch 13 to 15. After epoch 15, it was a steady line and the training accuracy reached 100%.

For the validation accuracy:

It starts around 20%, then steadily increases from epoch 0 to 5. From epochs 5 to 22, it fluctuates between 50% and 70% of validation accuracy. After epoch 22, it becomes a steady line and the validation accuracy reaches 73.85%.

```
[24] plt.figure(figsize=(10, 10))
     plt.subplot(2, 1, 1)
     plt.plot(acc, label='Training Accuracy')
     plt.plot(val_acc, label='Validation Accuracy')
     plt.legend(loc='lower right')
     plt.ylabel('Accuracy')
     #plt.ylim([min(plt.ylim()),1])
     plt.title('Training and Validation Accuracy')
     plt.show()
```

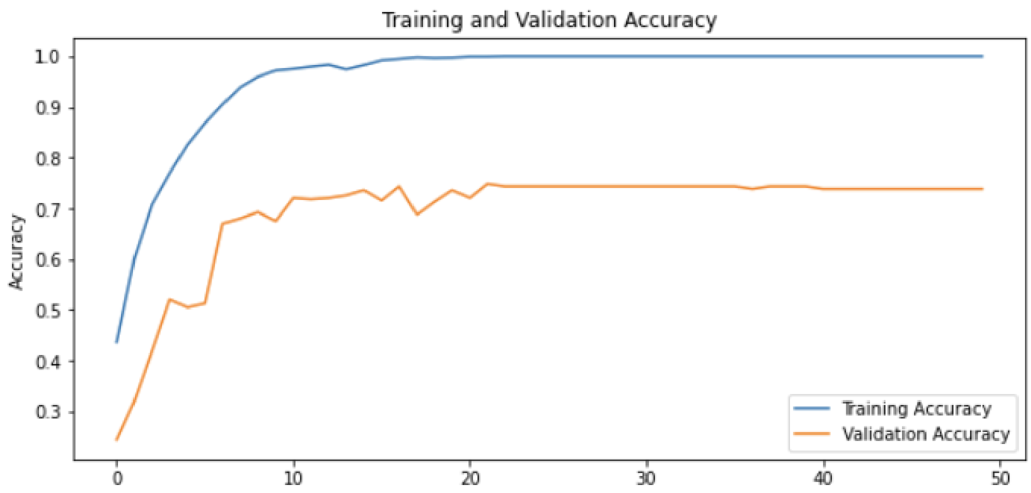

Figure 4.1.1 Training and Validation Accuracy

For the training loss:

It starts around 150% training loss, then slowly decreases from epoch 0 to 10. After epoch 10 to epoch 50, it becomes a steady line and the training loss reaches 0%.

For the validation loss:

It starts around 200% validation loss, then fluctuates between 200% and 600% of validation loss from epoch 0 to 22. Then, it steadily increases starting at epoch 22 until the last epoch, reaching a validation loss of 932.23%.

```
[25] plt.figure(figsize=(10, 10))
     plt.subplot(2, 1, 2)
     plt.plot(loss, label='Training Loss')
     plt.plot(val_loss, label='Validation Loss')
     plt.legend(loc='upper right')
     plt.ylabel('Cross Entropy')
     #plt.ylim([0,1.0])
     plt.title('Training and Validation Loss')
     plt.xlabel('epoch')
     plt.show()
```

Figure 4.1.2 Training and Validation Loss

3.2. Transfer Learning (TL)

For the transfer learning it will be done using three different architecture models which are MobileNetV2, VGG19, and ResNet101V2.

3.2.1. MobileNetV2

MobileNetV2 is a convolutional neural network architecture for image classification that was developed by Google.

For transfer learning it's using tensorflow.keras.applications to import MobileNetV2. Keras applications are deep learning models that are made available alongside pre-trained weights. Preprocess input is a tensor or Numpy array encoding a batch of images.

For the training accuracy:

It starts around 52%, then sharply increases from epoch 0 to 2. In epochs 2 to 5, it steadily increases. From epoch 5 to epoch 18, it fluctuates between 85% and 98%. Then between epochs 18 to 50, it becomes a steady line and the training accuracy reaches 100%.

For the validation accuracy:

It starts around 20%, then sharply increases from epoch 0 to 3. From epoch 3 to 6, it steadily increases. From epoch 6 to epoch 50, it fluctuates between 85% and 98%. Reaching validation accuracy of 75.89 % at epoch 50.

```
[ ]  plt.figure(figsize=(10, 10))
     plt.subplot(2, 1, 1)
     plt.plot(acc, label='Training Accuracy')
     plt.plot(val_acc, label='Validation Accuracy')
     plt.legend(loc='lower right')
     plt.ylabel('Accuracy')
     #plt.ylim([min(plt.ylim()),1])
     plt.title('Training and Validation Accuracy')
     plt.show()
```

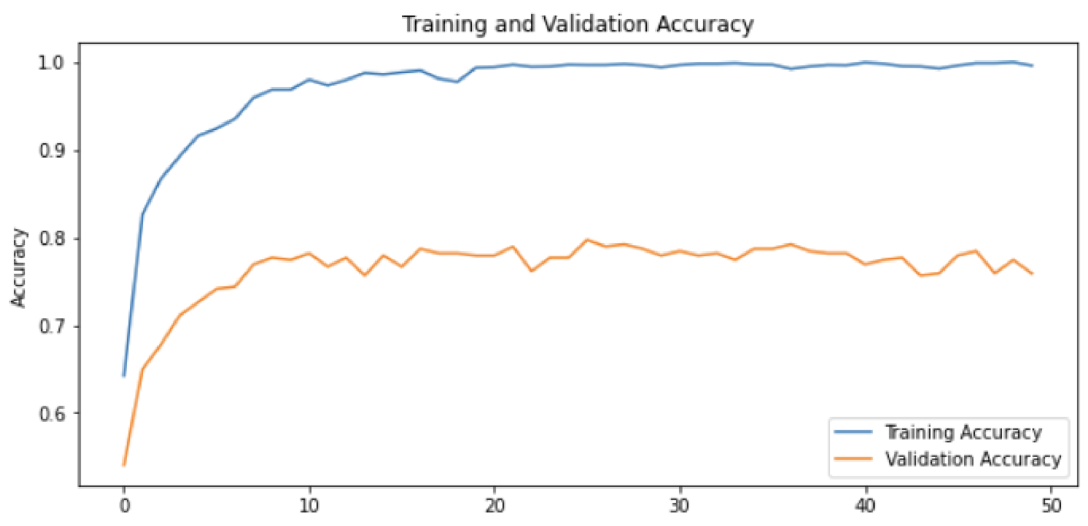

Figure 4.2.1.1 Training and Validation Accuracy

For the training loss:

It starts around 98% and steadily decreases until it reaches 0% to epoch 10. From epoch 10 to 50, the training loss is a straight line at 0% of training loss.

For the validation loss:

The validation loss fluctuates between 150% to 430% from epoch 0 to 50. Reaching validation loss of 373.42% at epoch 50.

```
[ ] plt.figure(figsize=(10, 10))
    plt.subplot(2, 1, 2)
    plt.plot(loss, label='Training Loss')
    plt.plot(val_loss, label='Validation Loss')
    plt.legend(loc='upper right')
    plt.ylabel('Cross Entropy')
    #plt.ylim([0,1.0])
    plt.title('Training and Validation Loss')
    plt.xlabel('epoch')
    plt.show()
```

Figure 4.2.1.2 Training and Validation Loss

3.2.2. VGG19

VGG19 is a convolutional neural network model trained on the ImageNet dataset. This model and its variants are used for image classification and other vision tasks. The "19" in the name refers to the number of weight layers in the network. VGG19 consists of 19 weight layers, which include 16 convolutional layers and three fully-connected layers. The convolutional layers are arranged in a series of convolutional blocks. Each block contains two or three convolutional layers and a max pooling layer. The fully-connected layers follow the convolutional layers and are used for classification.

Using tensorflow.keras.applications to import VGG19 architecture as seen in (*Figure 4.2.2.1*), which is a convolutional neural network that is 19 layers deep.

For the training accuracy:

It starts around 43%, then sharply increases from epoch 0 to 8. From epoch 8 to epoch 14, it slowly increases, then decreases from epoch 13 to 15. After epoch 15, it was a steady line and the training accuracy reached 100%.

For the validation accuracy:

From epoch 0 to 1, the validation accuracy sharply increased, reaching around 62%. Then, there was a steady increase from epoch 1 to 2, reaching 65%. From epochs 2 to 6, it sharply increased to 75% of validation accuracy. Then, from epoch 6 to 50, it fluctuated between 75% and 78%, reaching a validation accuracy of 77.92% at epoch 50.

```
[26] plt.figure(figsize=(10, 10))
     plt.subplot(2, 1, 1)
     plt.plot(acc, label='Training Accuracy')
     plt.plot(val_acc, label='Validation Accuracy')
     plt.legend(loc='lower right')
     plt.ylabel('Accuracy')
     #plt.ylim([min(plt.ylim()),1])
     plt.title('Training and Validation Accuracy')
     plt.show()
```

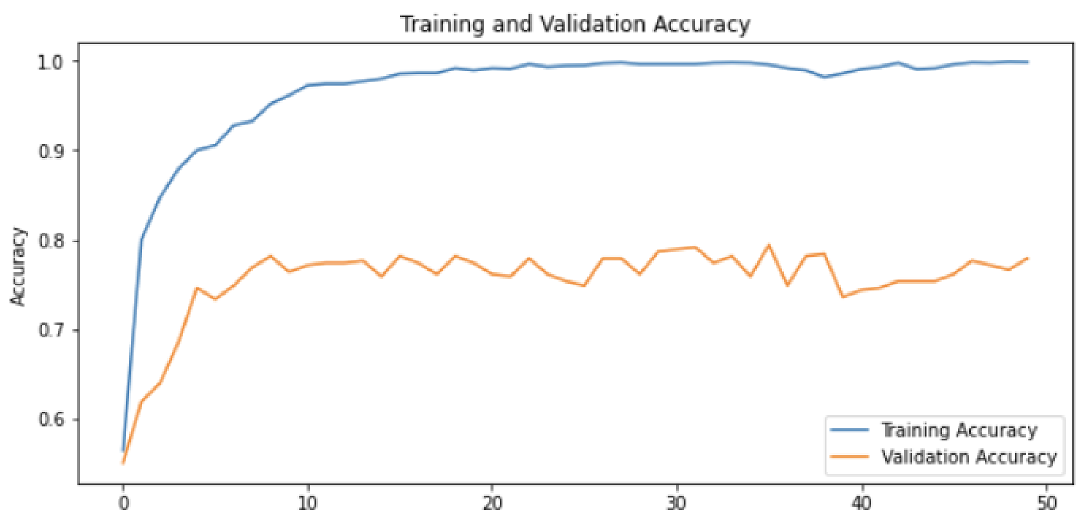

Figure 4.2.2.1 Training and Validation Accuracy

For the training loss:

It starts around 149% and sharply decreases to epoch 2. Then, from epochs 2 to 10, it steadily decreases until it reaches 0%. From epoch 10 to 50, the training loss is a straight line at 0% training loss.

For the validation loss:

The validation loss fluctuates between 120% and 340% from epoch 0 to 50, reaching a validation loss of 298.78% at epoch 50.

```
[27] plt.figure(figsize=(10, 10))
     plt.subplot(2, 1, 2)
     plt.plot(loss, label='Training Loss')
     plt.plot(val_loss, label='Validation Loss')
     plt.legend(loc='upper right')
     plt.ylabel('Cross Entropy')
     #plt.ylim([0,1.0])
     plt.title('Training and Validation Loss')
     plt.xlabel('epoch')
     plt.show()
```

Figure 4.2.2.2 Training and Validation Loss

3.2.3. ResNet101V2

ResNet101V2 is an image recognition convolutional neural network trained to distinguish objects in images. It is a subset of the ResNet101 network, which is a more in-depth version of the original ResNet network. The "V2" in the name alludes to an improved version of the original ResNet101 network.

For the training accuracy:

It starts around 43%, then sharply increases from epoch 0 to 1. From epoch 1 to epoch 8, it slowly increases. From epoch 8 to 35 it slowly fluctuates between 95% to 100%

of training accuracy. After epoch 35, it was a steady line and the training accuracy reached 100%.

For the validation accuracy:

From epoch 0 to 3, the validation accuracy sharply increased, reaching around 62%. Then, there was a steady increase from epoch 3 to 4, reaching 65%. From epochs 4 to 6, it sharply increased to 75% of validation accuracy. Then, from epoch 6 to 50, it fluctuated between 75% and 80%, reaching a validation accuracy of 79.19 % at epoch 50.

```
[ ]  plt.figure(figsize=(10, 10))
     plt.subplot(2, 1, 1)
     plt.plot(acc, label='Training Accuracy')
     plt.plot(val_acc, label='Validation Accuracy')
     plt.legend(loc='lower right')
     plt.ylabel('Accuracy')
     #plt.ylim([min(plt.ylim()),1])
     plt.title('Training and Validation Accuracy')
     plt.show()
```

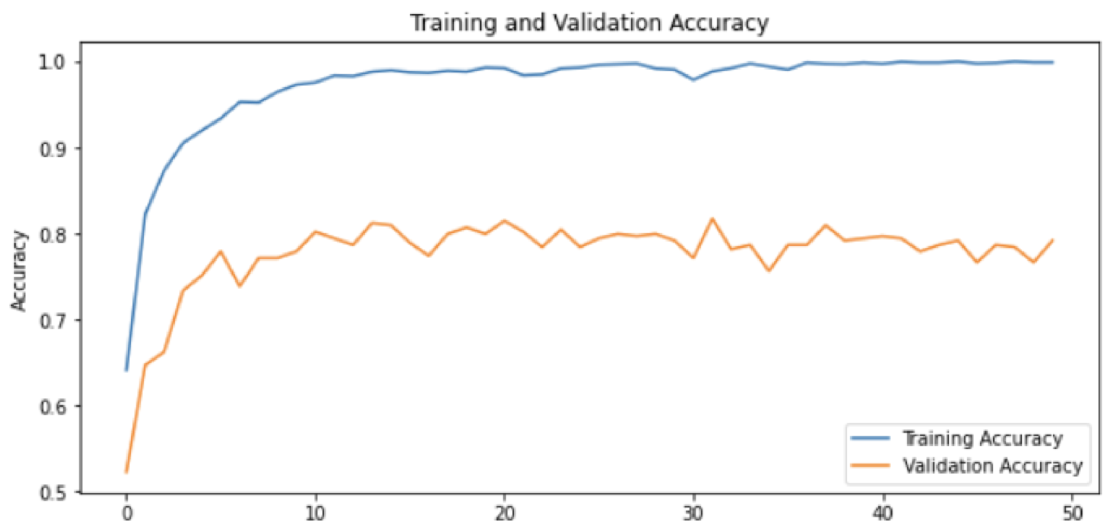

Figure 4.2.3.1 Training and Validation Accuracy

For the training loss:

It starts around 98% and sharply decreases to epoch 1. Then, from epochs 1 to 10, it steadily decreases until it reaches 0%. From epoch 10 to 50, the training loss is a straight line at 0% training loss.

For the validation loss:

It starts around 120% of the validation loss. From epoch 0 to 3, it steadily decreases to 99%. Then, from epoch 3 to 10, it slowly increases, reaching around 145%. At epochs

10 to 24, it steadily decreases to 118%. From epochs 24 to 50, it fluctuates between around 118% and 310% of validation loss, reaching a validation loss of 238.70% at epoch 50.

```
[ ] plt.figure(figsize=(10, 10))
    plt.subplot(2, 1, 2)
    plt.plot(loss, label='Training Loss')
    plt.plot(val_loss, label='Validation Loss')
    plt.legend(loc='upper right')
    plt.ylabel('Cross Entropy')
    #plt.ylim([0,1.0])
    plt.title('Training and Validation Loss')
    plt.xlabel('epoch')
    plt.show()
```

Figure 4.2.3.2 Training and Validation Loss

3.3. Comparison Between Models

All architecture models used in transfer learning were trained using 128 batches, with image sizes of (224,224) and 50 epochs. The same applies to convolutional neural networks. As illustrated in Table 4.3.1, the convolutional neural network's accuracy percentage (73.85%) is lower than that of all transfer learning architecture models, and the loss percentage (932.23%) is significantly higher than that of all transfer learning architecture models. Comparing all transfer learning architecture models, ResNet101V2 has the best accuracy rate (79.18%) and the lowest loss rate (238.77%), followed by VGG19, and finally, MoblieNetV2 has the lowest accuracy rate (75.88%) and the highest loss rate (373.42%).

	Transfer Learning		
Architecture	MobileNetV2	VGG19	ResNet101V2

Model			
Validation Accuracy (%)	75.88	77.91	79.18
Validation Loss (%)	373.42	298.77	238.77
	Convolutional Neural Network (CNN)		
Validation Accuracy (%)	73.85		
Validation Loss (%)	932.23		

Table 4.3.1 Comparison Table

4. Initialization, Training, and Tuning of Hyperparameters

4.1. Hyperparameters

Under this section includes the initialization and tuning of several hyperparameters related to transfer learning and convolutional neural networks. Training the models using different numbers of batches, two optimizers, the image size set to (224,224) and the number of epochs set to 50.

The two optimizers are Adam and RMSprop:

The **Adam** optimizer is used to accelerate the gradient descent algorithm by taking into consideration the 'exponentially weighted average' of the gradients. It uses the averages to make the algorithm converge toward the minima at a faster pace.

In **RMSprop** it restricts the oscillations in the vertical direction. Therefore, it increases the learning rate and the algorithm can take larger steps in the horizontal direction converging faster.

4.2. Conventional Neural Networks (CNN)

Batch Number	Number of Epochs	Image Size	Optimizer	Testing Results	
				Accuracy (%)	Loss (%)
128	50	(224,224)	Adam	73.85	932.23
64	50	(224,224)	Adam	74.11	910.35
128	50	(224,224)	RMSprop	71.83	1541.56

64	50	(224,224)	RMSprop	75.38	1384.89

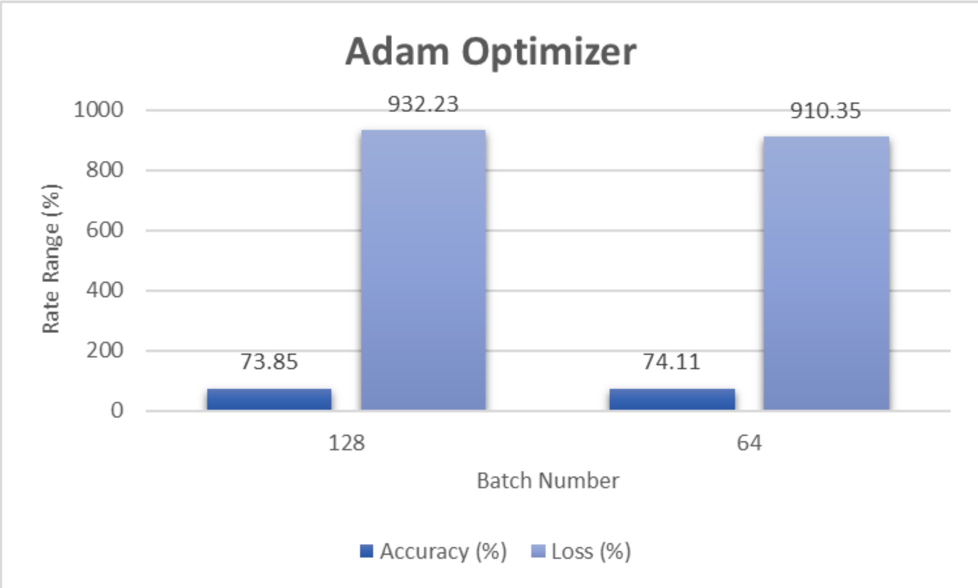

Figure 5.1.1 CNN Adam Optimizer Results

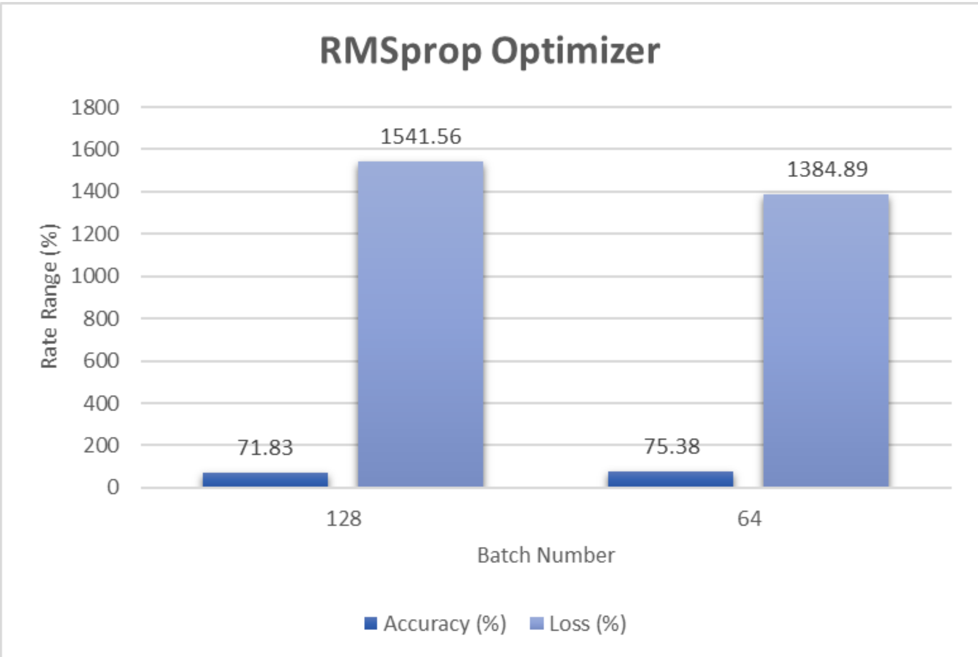

Figure 5.1.2 CNN RMSprop Optimizer Results

4.3. Transfer Learning (TL)

4.3.1. MobileNetV2

Batch	Number of			Testing Results	

Number	Epochs	Image Size	Optimizer	Accuracy (%)	Loss (%)
128	50	(224,224)	Adam	75.89	373.42
64	50	(224,224)	Adam	76.90	370.42
128	50	(224,224)	RMSprop	77.16	383.13
64	50	(224,224)	RMSprop	76.90	562.44

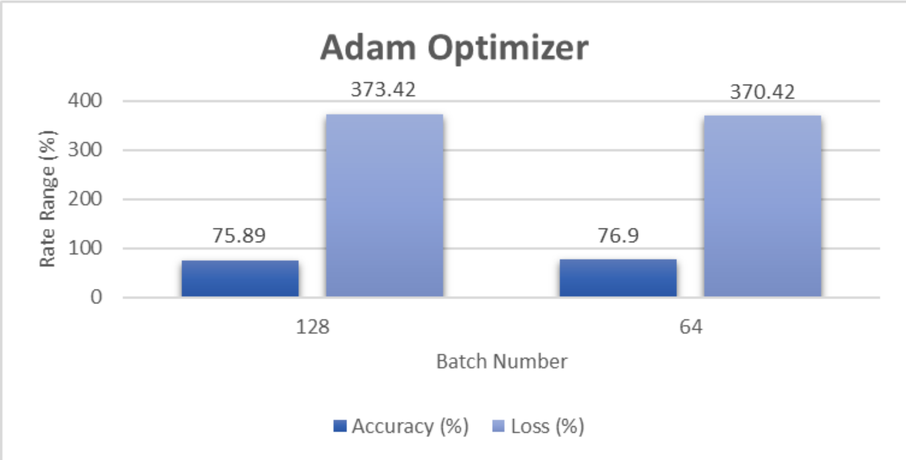

Figure 5.2.1.1 MobileNetV2 Adam Optimizer Results

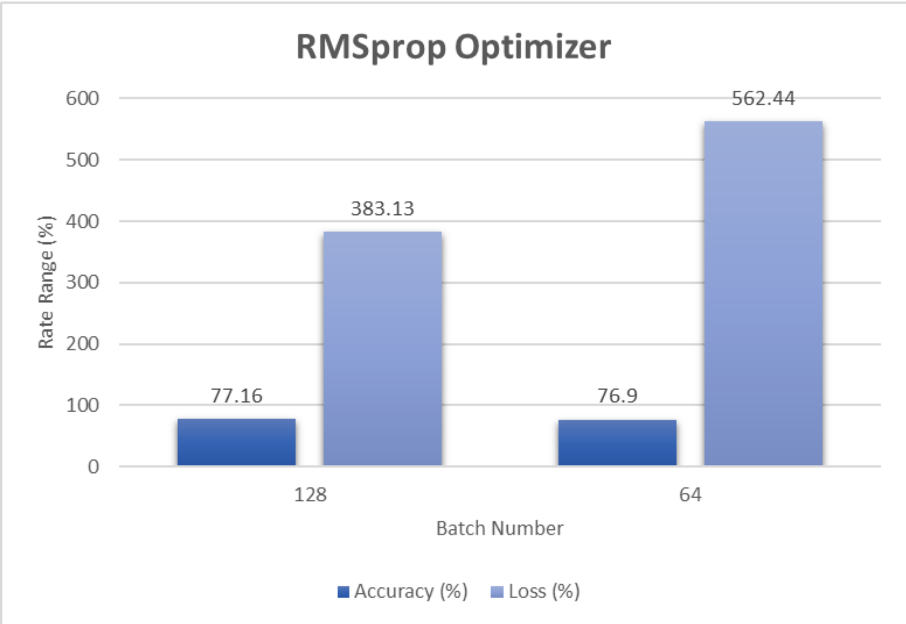

Figure 5.2.1.2 MobileNetV2 RMSprop Optimizer Results

4.3.2. VGG19

Batch Number	Number of Epochs	Image Size	Optimizer	Testing Results	
				Accuracy (%)	Loss (%)
128	50	(224,224)	Adam	77.92	298.78
64	50	(224,224)	Adam	75.38	415.14
128	50	(224,224)	RMSprop	76.40	339.28
64	50	(224,224)	RMSprop	74.62	505.32

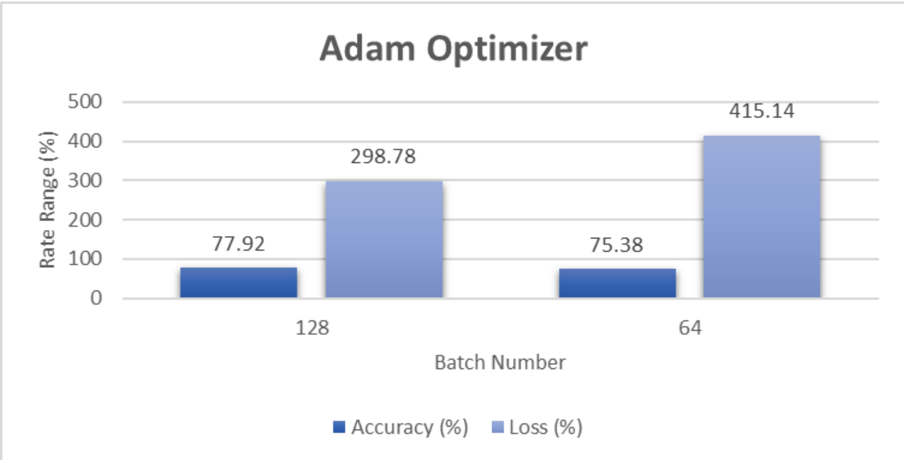

Figure 5.2.2.1 VGG19 Adam Optimizer Results

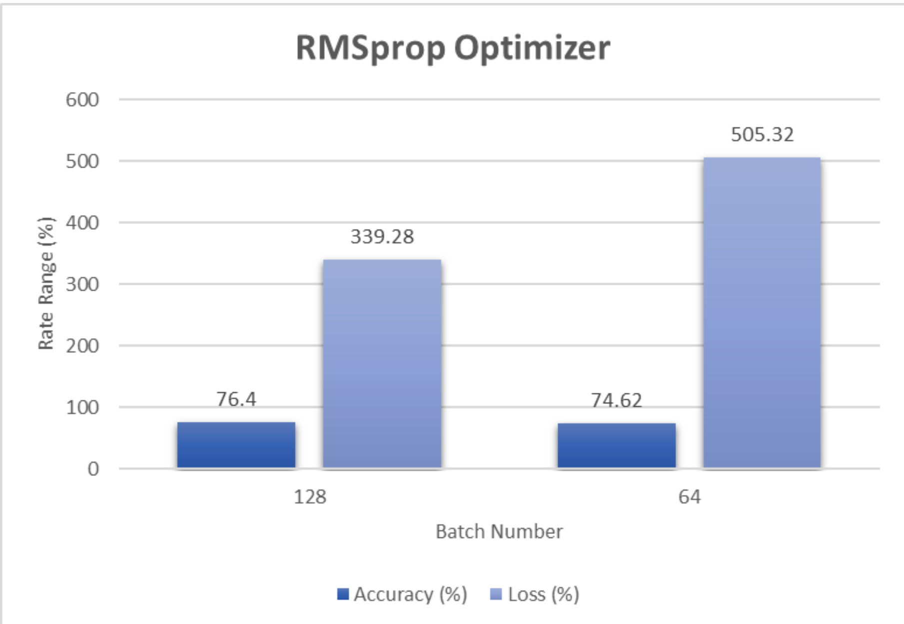

Figure 5.2.2.2 VGG19 RMSprop Optimizer Results

4.3.3. ResNet101V2

Batch Number	Number of Epochs	Image Size	Optimizer	Testing Results	
				Accuracy (%)	Loss (%)
128	50	(224,224)	Adam	79.19	238.70
64	50	(224,224)	Adam	78.17	255.89
128	50	(224,224)	RMSprop	81.73	248.79
64	50	(224,224)	RMSprop	78.43	358.60

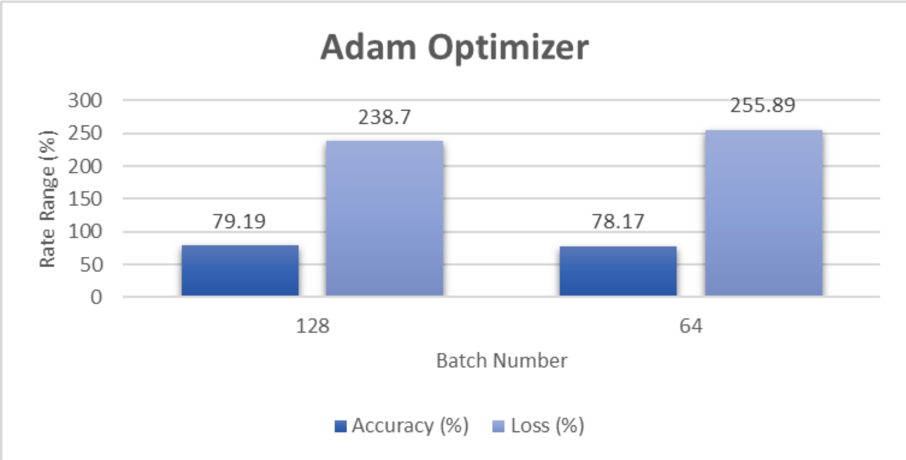

Figure 5.2.3.1 VGG19 Adam Optimizer Results

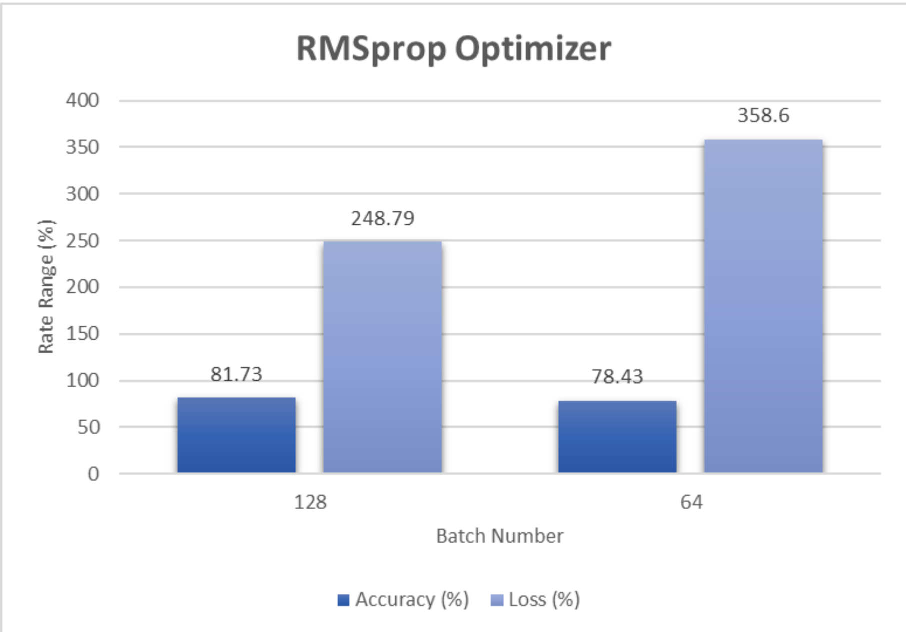

Figure 5.2.3.2 VGG19 RMSprop Optimizer Results

5. Performance evaluation, comparison, and discussion of the results

Comparison of the Machine Learning Models

The comparison of Transfer learning and Convolutional Neural Networks models, in the second paragraph, describes the comparison between all three architecture models of transfer learning, and in the third paragraph, it describes the comparison between the transfer learning and convolutional neural networks models.

As illustrated in Figure 6.1, shows the accuracy and loss rates of all three architectures using different numbers of batches. Using the Adam optimizer with a batch number of 128 with three different architecture models, the ResNet101V2 architecture model has the highest rate of accuracy 79.19%, and the lowest loss rate 238.7%. When the batch number was halved, the ResNet101V2 had the highest accuracy rate at 78.17% and the lowest loss rate at 255.89%.

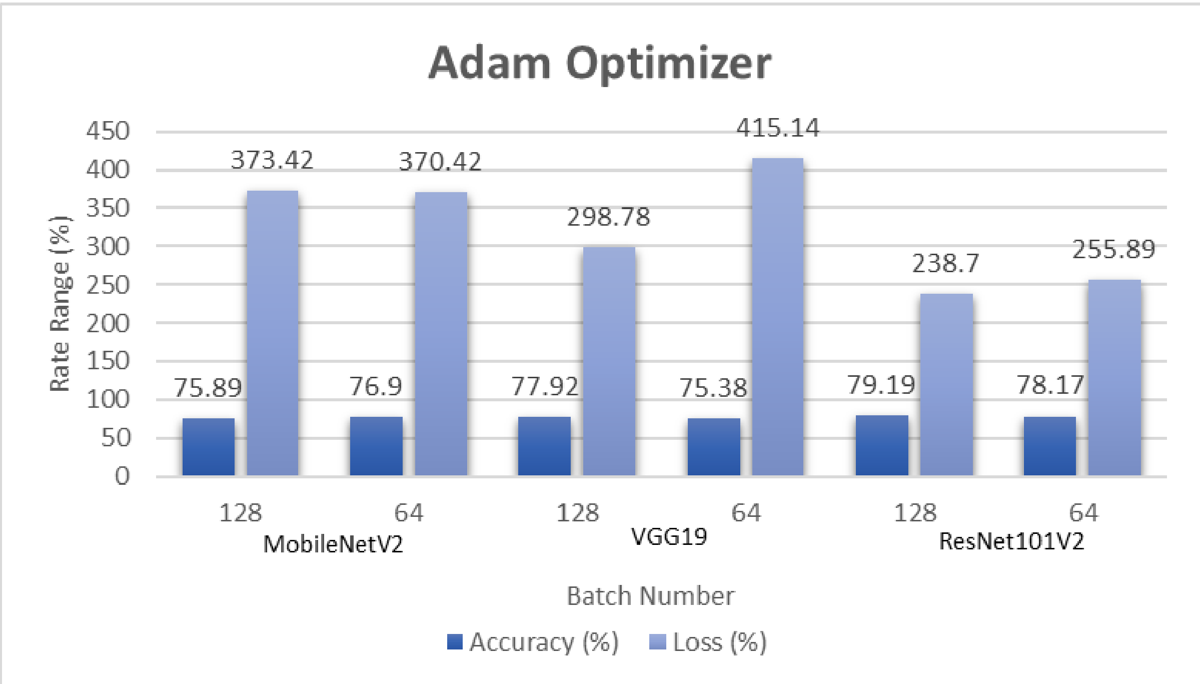

Figure 6.1 Transfer Learning Adam Optimizer

When the optimizer was changed to RMSprop with a number of 128 batches, the ResNet101V2 had the highest accuracy rate of 81.73% and the lowest loss rate of 248.79%. When the batch number was decreased to 64, the ResNet101V2 had the highest accuracy rate 78.43%, and the lowest loss rate 358.6%.

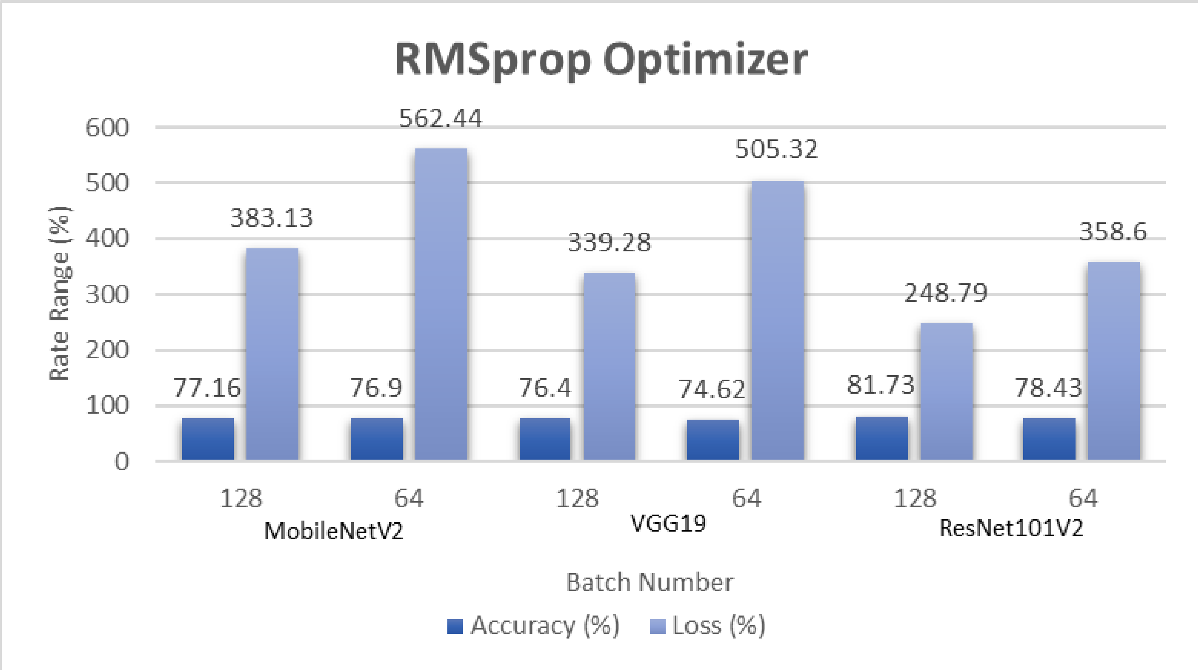

Figure 6.2 Transfer Learning RMSprop Optimizer

Using the CNN model with the number of 128 batches, the Adam optimizer had a higher accuracy rate of 73.85% and a lower loss rate of 932.23%. When the number of batches was halved, the RMSprop optimizer had a higher accuracy rate of 75.38% and the Adam optimizer had a lower loss rate of 910.35%.

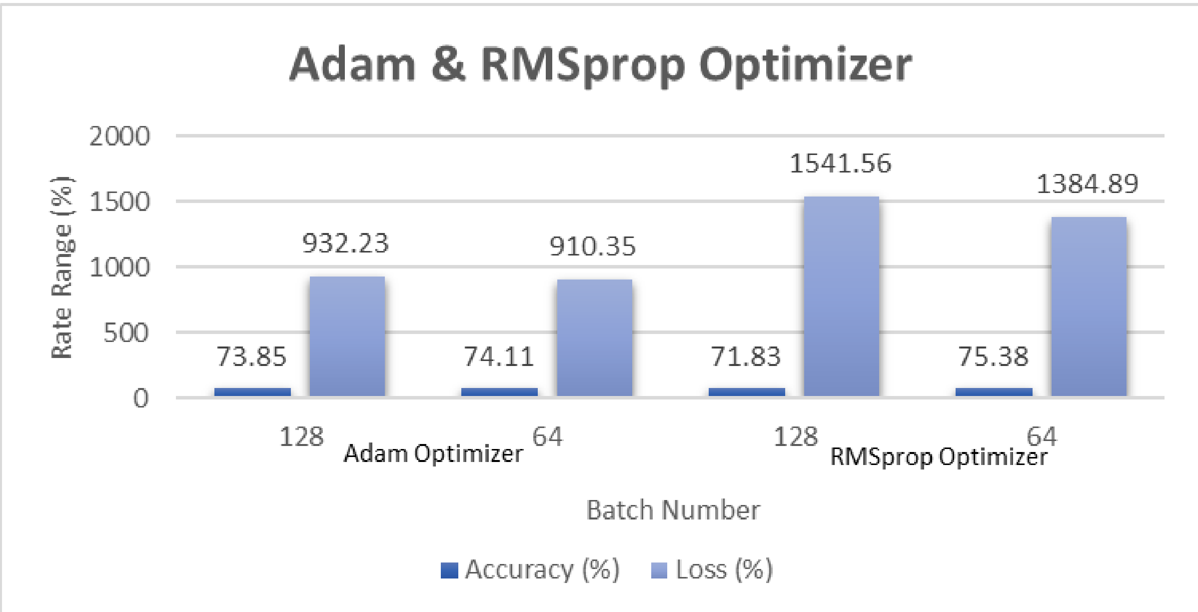

Figure 6.3 CNN Adam & RMSprop Optimizer

Lastly, comparing transfer learning and convolutional neural networks, transfer learning had a much better accuracy and loss rate compared to CNN. The accuracy rate of CNN was similar to the transfer learning accuracy rate, and the loss rate was much higher than the transfer learning loss rate. However, in transfer learning, three different architecture models were used with two different numbers of batches and two types of optimizers. Out of the three architecture models, the ResNet101V2 using the RMSprop optimizer and the number of 128 batches had the highest accuracy rate at 81.73% with a loss rate of 248.79%. Using ResNet101V2 with the Adam optimizer and the number of 128 batches had the lowest loss rate 238.7% with an accuracy rate of 93.99%.

To achieve a higher accuracy rate, use the ResNet101V2 architecture model with the RMSprop optimizer and a number of 128 batches. However, to achieve a lower loss rate, use the ResNet101V2 architecture model with the Adam optimizer and a number of 128 batches.

Overall, the loss rate for CNN and transfer learning is so high that could cause due to:

- The lack of data: The total of images in the train directory was 2870 and in the tests was 394. To address this, more data can be collected or data augmentation can be used.
- Overfitting: This occurs when a model is trained too well on the training data, resulting in poor performance on the validation or test data. To decrease the validation loss caused by overfitting, techniques like regularization, dropout, and early stopping can be used.

In summary, to decrease the validation loss, it is important to use techniques like regularization, dropout, and early stopping, to try different architectures and hyperparameters, and to ensure that there is enough data.

6. Conclusion and Future Enhancements

In conclusion, the research project aimed to detect brain tumors using CNNs and transfer learning. It found that transfer learning had better accuracy and loss rate compared to conventional neural networks. The ResNet101V2 model with RMSprop optimizer and batch number of 128 had the highest accuracy and lowest loss rate. The high loss rate may be due to a lack of data and overfitting, which can be addressed by collecting more data, using data augmentation, and applying regularization, dropout, and early stopping. To decrease validation loss, different architectures and hyperparameters should be tried.

As for future enhancements, it is suggested to try different datasets or other pre-trained models for transfer learning. Another possible enhancement is to try ensembling multiple models to improve the overall performance and to also try fine-tuning the pre-trained models.

References

[1] Amin, J., Sharif, M., Yasmin, M., & Fernandes, S. L. (2020, November). A distinctive approach in brain tumor detection and classification using MRI. *Pattern Recognition Letters*, *139*, 118–127. https://doi.org/10.1016/j.patrec.2017.10.036

[2] Keerthana, A., Kavin Kumar, B., Akshaya, K., & Kamalraj, S. (2021). Brain Tumour Detection Using Machine Learning Algorithm. Journal of Physics: Conference Series, 1937(1), 012008. https://doi.org/10.1088/1742-6596/1937/1/012008

[3] Liu, Y., Carpenter, A., Yuan, H., Zhou, Z., Zalutsky, M., Vaidyanathan, G., Yan, H., & Vo-Dinh, T. (2016, July 15). Abstract 4213: Gold nanostar as theranostic probe for brain tumor sensitive PET-optical imaging and image-guided specific photothermal therapy. *Cancer Research*, *76*(14_Supplement), 4213–4213. https://doi.org/10.1158/1538-7445.am2016-4213

[4] Morgan, G. W., Foster, K., Healy, B., Opie, C., & Huynh, V. (2018). Improving Health and Cancer Services in Low-Resource Countries to Attain the Sustainable Development Goals Target 3.4 for Noncommunicable Diseases. *Journal of Global Oncology*, *4*, 1–11. https://doi.org/10.1200/jgo.18.00185

[5] *MRI*. (2021, September 4). Mayoclinic. https://www.mayoclinic.org/tests-procedures/mri/about/pac-20384768