MSc in Data Science

Deep Learning

Medical Image Analysis

Nikolaos-Marios Tsarouchas

Marios Vottas

Contents

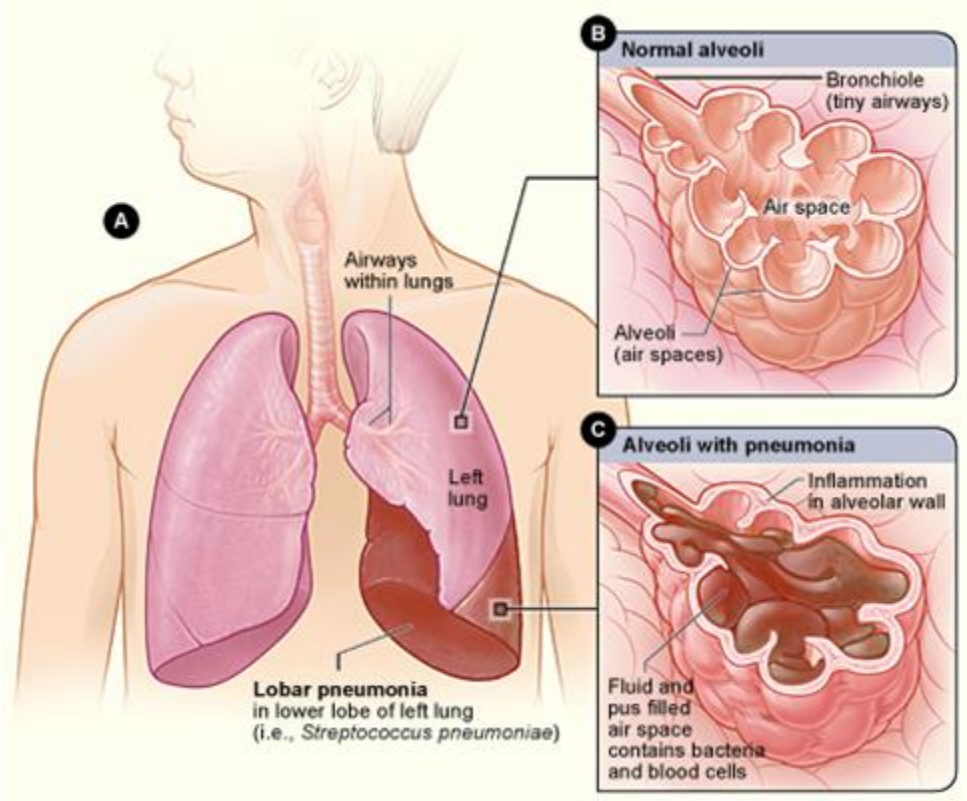
Abstract

Early identification of a patient's critical condition is vital in order to receive as soon as possible the necessary treatment and increase the possibilities of survival. One common critical condition is Pneumonia where the alveoli are filled with fluid and a person's life is in threat. Another rare disease is brain tumour which has high rates of fatality. The diagnosis for both case is made by Xray's, MRI's and blood test. The research of computer vision, imaging processing and pattern recognition has made substantial progress during the past several decades. Also, medical imaging has attracted increasing attention in recent years due to its vital component in healthcare applications. Investigators have published a wealth of basic science and data documenting the progress and healthcare application on medical imaging. In our study we try to classify the condition of pneumonia from annotated Xray's using several convolution neural networks and deeper convolution neural network architecture like resnet and densenet using transfer learning techniques. Finally, for the brain tumour segmentation problem we developed a unet architecture in order to identify the abnormality segments in the MRI's.

1. Pneumonia Classification

Pneumonia is a form of acute respiratory infection that affects the lungs. The lungs are made up of small sacs called alveoli, which fill with air when a healthy person breathes. When an individual has pneumonia, the alveoli are filled with pus and fluid, which makes breathing painful and limits oxygen intake. Pneumonia is the single largest infectious cause of death in children worldwide. Pneumonia killed 740.180 children under the age of 5 in 2019, accounting for 14% of all deaths of children under five years old but 22% of all deaths in children aged 1 to 5. Pneumonia affects children and families everywhere, but deaths are highest in South Asia and sub-Saharan Africa. Children can be protected from pneumonia, it can be prevented with simple interventions, and treated with low-cost, low-tech medication and care. Pneumonia can be caused by numerous infectious agents such us viruses, bacteria and fungi.[1]

[1] Info was taken from WHO



The identification is possible by specialized doctors with the examination of chest Xrays, blood tests and of course the patient. Sometimes, it is very difficult to identify pneumonia and it is very important to do it as fast as possible in order to provide the necessary treatment. Computer vision and image classification is a new weapon into doctors' arsenal in order to prevent deterioration of patient's health condition.

Data Set

The dataset is organized into 3 folders (train, test, val) and contains subfolders for each image category (Pneumonia/Normal). There are 5,863 X-Ray images (JPEG) and 2 categories (Pneumonia/Normal).

Chest X-ray images (anterior-posterior) were selected from retrospective cohorts of pediatric patients of one to five years old from Guangzhou Women and Children's Medical Center, Guangzhou. All chest X-ray imaging was performed as part of patients' routine clinical care. For the analysis of chest x-ray images, all chest radiographs were initially screened for quality control by removing all low quality or unreadable scans. The diagnoses for the images were then graded

by two expert physicians before being cleared for training the AI system. In order to account for any grading errors, the evaluation set was also checked by a third expert.

Preprocessing

In order to use the images as input for the deep learning models, some preprocessing has to take place. First, the data are separated in train and test set as given from the initial file. Furthermore, the images are turned into batches so the model can be trained in small amounts of data each time. That is because the images take up a lot of memory, and as the number of images is large there is a danger of running out of memory. Moreover, the size of the images is changed as the original images are too large for deep learning models. If the images are very big, then the model has to learn many more parameters and as the number of parameters increases so does the complexity of the model. So, the image size that is used is 256 by 256. Also, the original images have RGB colour values, which are three parameters. This is changed to grayscale which is only one parameter. After the initial processing of the images additional transformations take place. Neural Networks have a tendency to perform when the input is normalized. With that said, both in training and test set normalization is performed. In addition, training set is separated in train and validation test. That is because it is a good practice in deep learning that the model does not see the test set until its final evaluation. So, all the tuning is tested with the validation test. In contrast to the test set, the training and validation test receive additional preprocessing. The images are getting horizontally flipped, vertically flipped, getting rotated and zoomed. The reason this data augmentation is taking place only in training set is that while the model is trained, if many images look alike there may be a chance of overfitting. With data augmentation this chance is significantly lowered. However, the test set must remain as is, because in a real world scenario, where this model is supposed to be deployed, the images will not be augmented. The last preprocessing method that is used in this dataset, is also in the training set. It is observed that the positive class examples are a lot more than the negative class examples. That is why the weights of each class was balanced. The weights are parameters that the models use and change in order to make better predictions.

Models

CNN

The Convolutional Neural Networks that are built are a result of multiple tries. The architecture of these models is developed specifically for this project, and while they may seem the same, all of them have some small differences between them. The first model has a pattern of a convolutional layer followed by a batch normalization followed by a max pooling layer. This pattern is repeated twice and then a dropout layer is inserted. This architecture is repeated one more time. Then a flatten layer links the CNN with two fully connected layers, followed by one dropout layer and three more fully connected layers. This is the entire architecture of the first model.

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 256, 256, 32)	320
batch_normalization (BatchNo	(None, 256, 256, 32)	128
max_pooling2d (MaxPooling2D)	(None, 128, 128, 32)	0
conv2d_1 (Conv2D)	(None, 128, 128, 64)	18496
batch_normalization_1 (Batch	(None, 128, 128, 64)	256
max_pooling2d_1 (MaxPooling2	(None, 64, 64, 64)	0
dropout (Dropout)	(None, 64, 64, 64)	0
conv2d_2 (Conv2D)	(None, 64, 64, 128)	73856
batch_normalization_2 (Batch	(None, 64, 64, 128)	512
max_pooling2d_2 (MaxPooling2	(None, 32, 32, 128)	0
conv2d_3 (Conv2D)	(None, 32, 32, 128)	147584
batch_normalization_3 (Batch	(None, 32, 32, 128)	512
max_pooling2d_3 (MaxPooling2	(None, 16, 16, 128)	0
dropout_1 (Dropout)	(None, 16, 16, 128)	0
flatten (Flatten)	(None, 32768)	0
dense (Dense)	(None, 256)	8388864
dense_1 (Dense)	(None, 128)	32896
dropout_2 (Dropout)	(None, 128)	0
dense_2 (Dense)	(None, 64)	8256
dense_3 (Dense)	(None, 32)	2080
dense_4 (Dense)	(None, 1)	33

Total params: 8,673,793
Trainable params: 8,673,089
Non-trainable params: 704

The same architecture with a slight increase in convolutional layers is followed in the other two CNNs. Since the models have been described, now will be explained what each layer means and what is its purpose in a CNN. The convolutional layer is the main building block of a CNN. It contains a set of filters called kernels. These filters are parameters that will be learned throughout

the training. The batch normalization is a technique for training very deep neural networks that normalizes the contributions to a layer for every mini-batch. This has the impact of settling the learning process and drastically decreasing the number of training epochs required to train deep neural networks. The max pooling layer is a pooling operation that selects the maximum element from the region of the feature map covered by the filter. Thus, the output after max-pooling layer would be a feature map containing the most prominent features of the previous feature map. The dropout layer randomly sets input units to 0 with a frequency of rate at each step during training time, which helps prevent overfitting. Flatten layer is used to make the multidimensional input one-dimensional, commonly used in the transition from the convolution layer to the full connected layer as seen in this CNN too. The dense layer is simple layer of neurons in which each neuron receives input from all the neurons of previous layer. Furthermore, besides building the models, compilation needs to take place. RMSprop was set as the optimizer for all the models, Adam is another very good optimizer for image classification tasks but for this project RMSprop was giving better results. The loss function used is binary cross entropy and the metric used for the performance of the models is accuracy. Moreover, callbacks were set. The learning rate to be reduced if there isn't enough improvement, the best model to be saved each time and if there is no improvement the fitting is stopped in order to save time and resources.

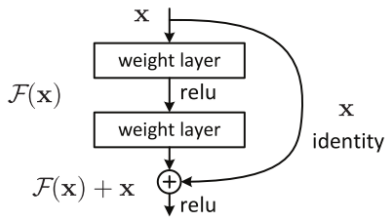
DenseNet

DenseNet (Dense Convolutional Network) is an architecture that focuses on making the deep learning networks go even deeper, but at the same time making them more efficient to train, by using shorter connections between the layers. DenseNet is a convolutional neural network where each layer is connected to all other layers that are deeper in the network. This is done to enable maximum information flow between the layers of the network. To preserve the feed-forward nature, each layer obtains inputs from all the previous layers and passes on its own feature maps to all the layers which will come after it. DenseNet consists of two important blocks other than the basic convolutional and pooling layers, the Dense blocks and the Transition layers. In this project, a DenseNet121 was built from scratch using TensorFlow code, it has 6,12,24,16 layers in the four dense blocks. Then the transition layer and a fully connected output layer especially designed for this task. Sigmoid activation was used in the output layer because it is a binary classification task.

ResNet

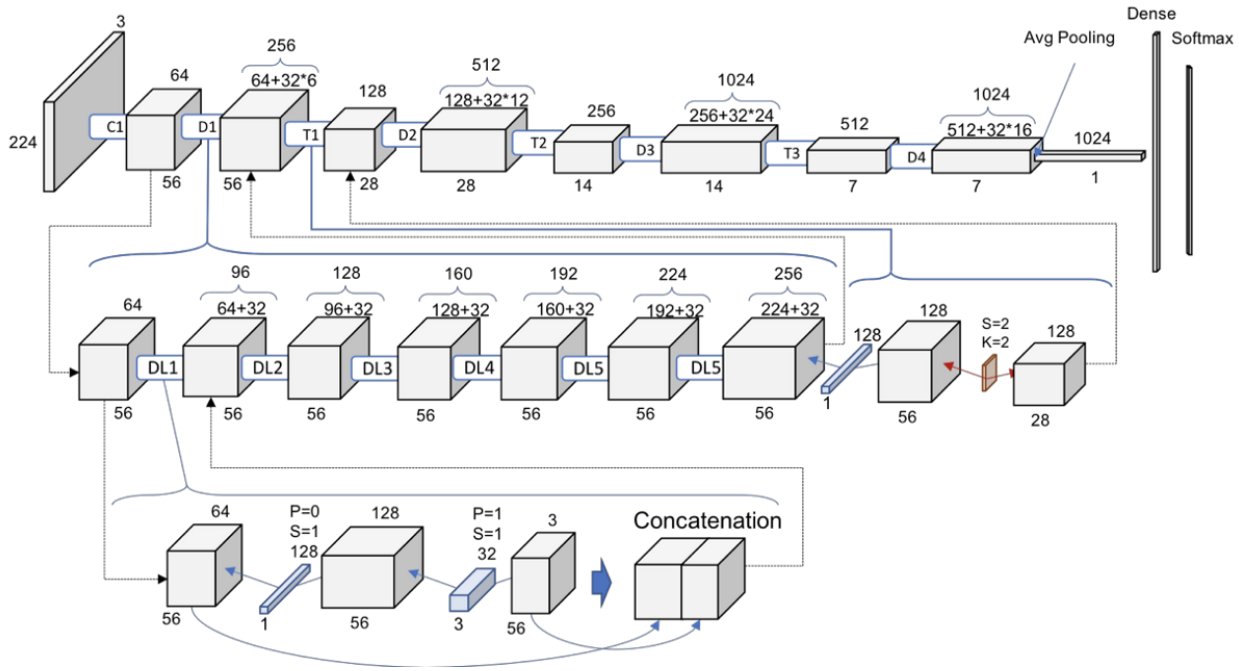
ResNet (Residual Neural Network) is a very deep feedforward neural network with hundreds of layers, much deeper than previous networks. However, increasing network depth does not work by simply stacking layers together. Deep networks are hard to train because of the notorious

vanishing gradient problem. As the gradient is back-propagated to earlier layers, repeated multiplication may make the gradient infinitively small. The birth of ResNet try to overcome the problem of vanishing gradients and create efficient very deep neural networks. The core idea of ResNet is introducing a so-called “identity shortcut connection” that skips one or more layers and with this way manages to deal with vanishing gradients. The image below shows how shortcut is performed.



Transfer Learning

Transfer learning is the application of previously trained models and is generally used to save time and resources from having to train multiple machine learning models from scratch to complete similar tasks. In the areas of machine learning that require high amounts of resources such as image categorization transfer learning is used. In addition to, all the models that have been custom built and trained, two pre trained models have been deployed. A ResNet152 and a DenseNet161.



DenseNet161 architecture

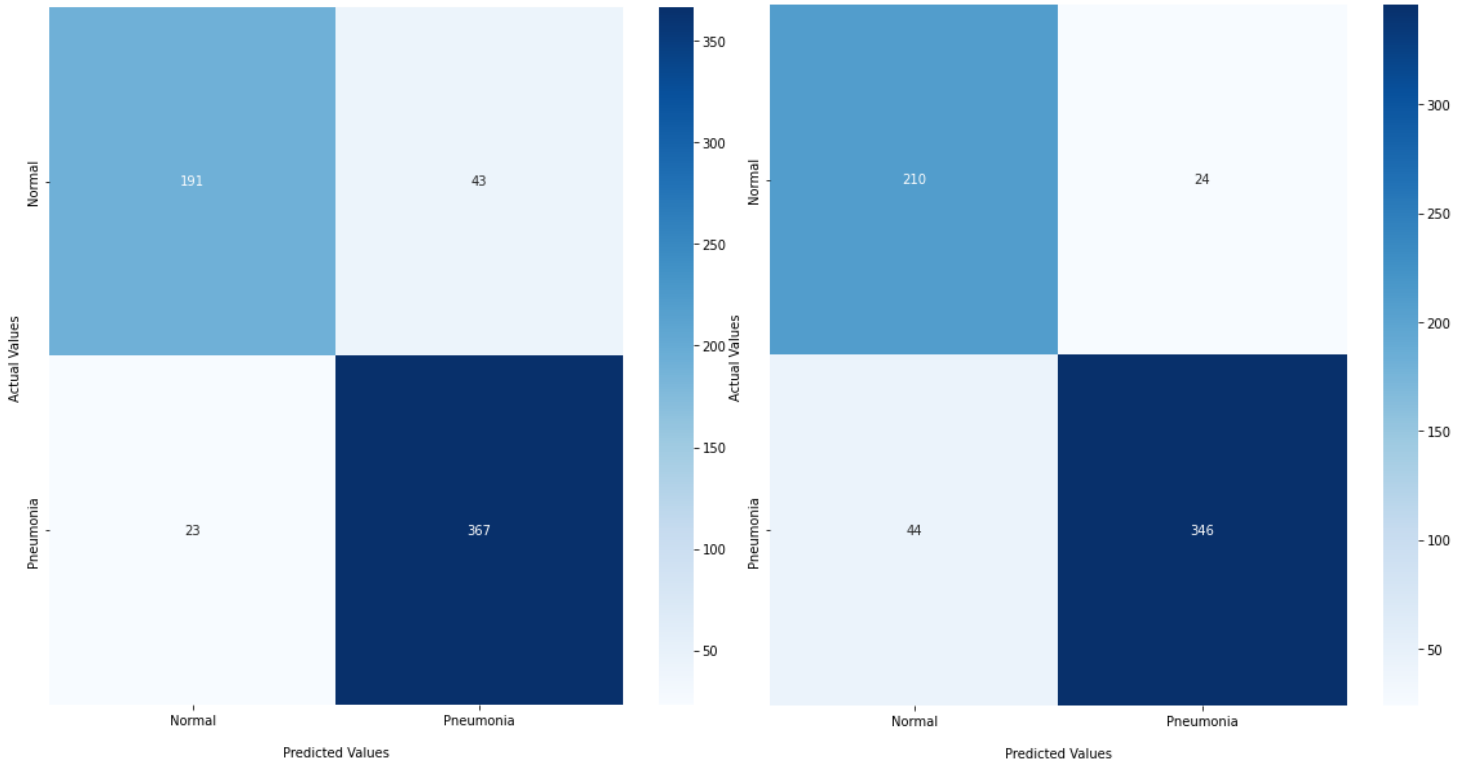
In the application of Transfer Learning instead of using randomly initial weights and train them we replaced with the pre trained ones in ImageNet. The update of those weights is being deactivated and the model is trained only for the fully connected layers. The update of the weights is being deactivated because the model could lose previously gained knowledge. This way, both of these models were used and deployed for this project.

Results

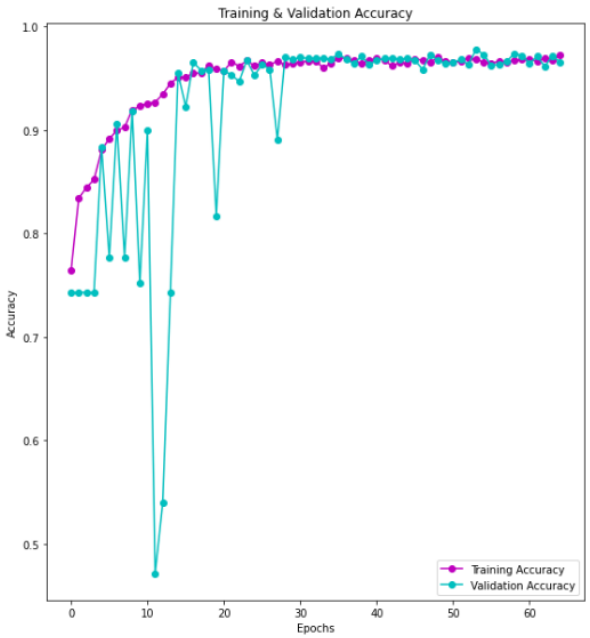
Each model is trained, validated and tested on the same data. The results of the models will be presented in the table below.

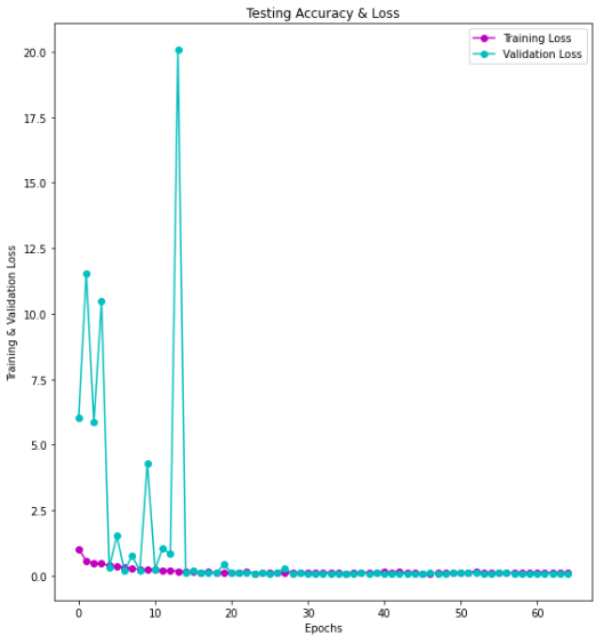
	CNN 1	CNN 2	CNN 3
Accuracy	0.89	0.89	0.88
Precision	0.90	0.94	0.93
Recall	0.94	0.89	0.88

It can be observed that model 1 and 2 have the best results. But one has a higher precision score and the other one has a higher recall score. In this particular task the first model is better. That is because since it's medical data recall is more important.



The left confusion matrix is from CNN1 and the right for CNN2. It can be seen that CNN1 which is considered better has less false negatives than CNN2. This means that if this model was used in production less people would be diagnosed healthy while having pneumonia. This is preferred rather than having diagnosed someone with pneumonia while being healthy. However, to reach these good results the models have to train. The process of training is not only upwards. The models could get worsen while time passes or they could remain stable and just waste resources. This is why callbacks were set to manage those situation, and the plots that follow will show how these callbacks worked in the training. When the model's loss was increasing, Reduce Learning Rate on Plateau callback was set off and managed to decrease the model's losses. Also, early stopping was set off when the model was no longer improving. That's why it trained for about 65 epochs and not 100 as it was original meant to train. These are the plot curves for CNN 1.



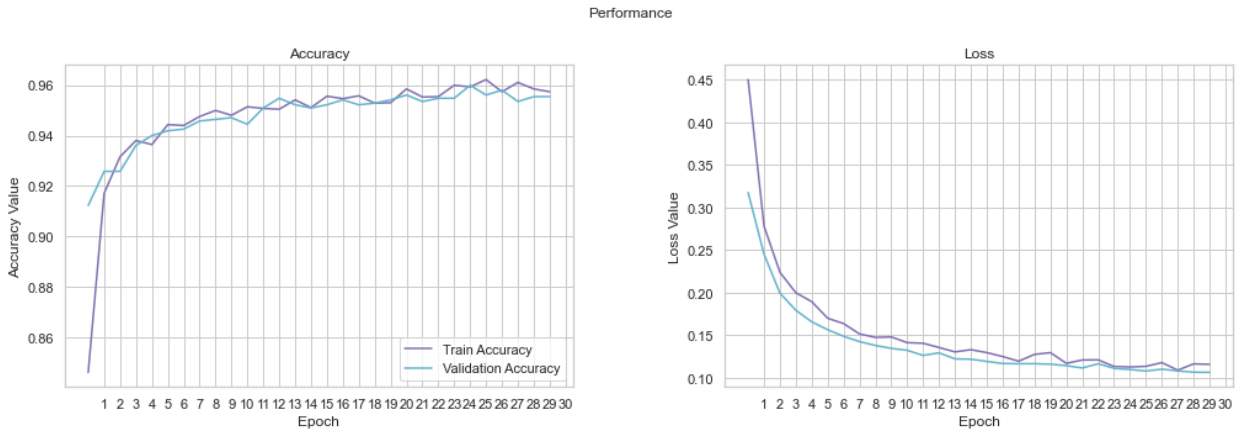


In this project two different DenseNet models where deployed. A custom built DenseNet121 and a pre trained DenseNet161 using transfer learning. Their results will be compared in the table below.

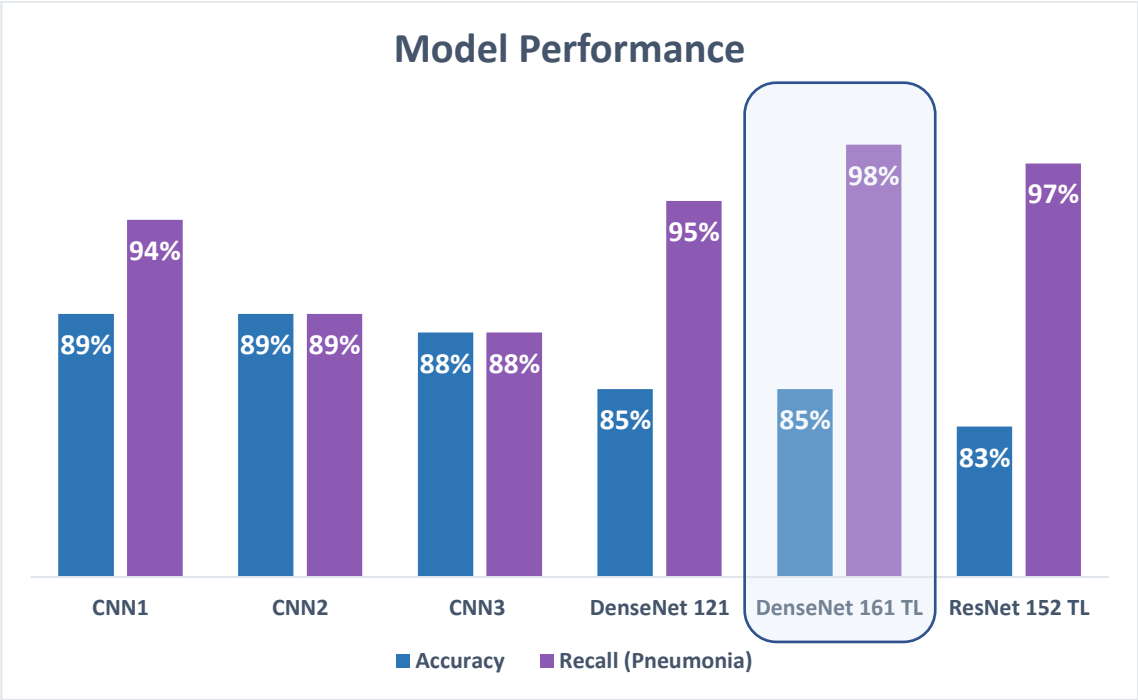
	DenseNet121	DenseNet161 TL
Validation Accuracy	0.949	0.963
Test Accuracy	0.854	0.846
Precision	0.840	0.810
Recall	0.950	0.980

As shown before, the Recall is the most important metric for this type of classification. As a result the transfer learning model is better but it is worth mentioning that the custom made DenseNet121 has excellent results despite it is trained with almost 5.000 images.

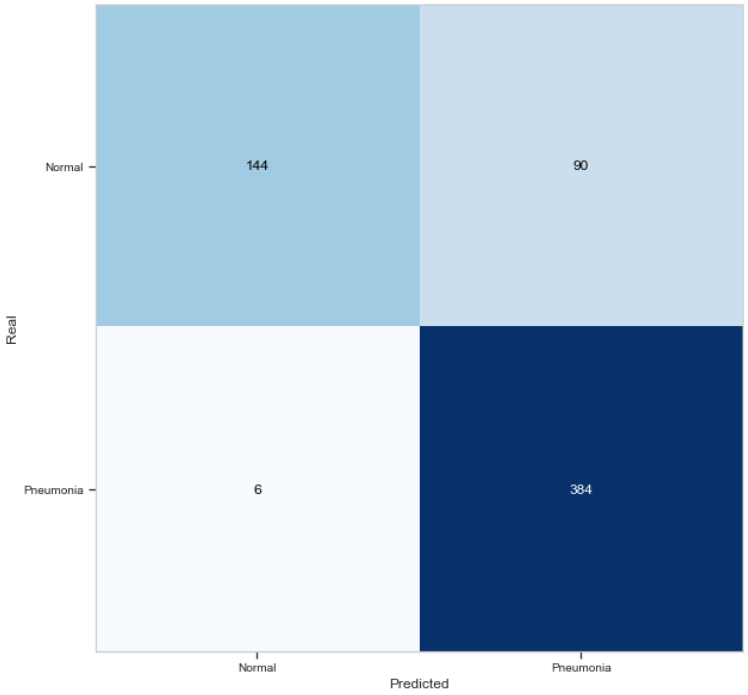
Although, the pre trained DenseNet has the best results, the other transfer learned model also returns very promising scores. Run only in 30 epochs, the model has an accuracy of 95% in the validation test and 82% on the test accuracy.



Despite the fact, that 82% might seem even lower than the DenseNet121 that is developed, it has the second-best recall score of 97%. With only 11 instances miss classified as a type II error. Which means that only 11 patients are predicted healthy while having pneumonia, this is the type of error that needs to be eliminated and ResNet152 has only 2 instances wrong more than the best model of this project. The comparison of the results can be shown together in the figure below:



Thus, as mentioned before the Densenet161 architecture with transfer learning performed better in the detection of Pneumonia. The confusion matrix is portrayed in the figure below:

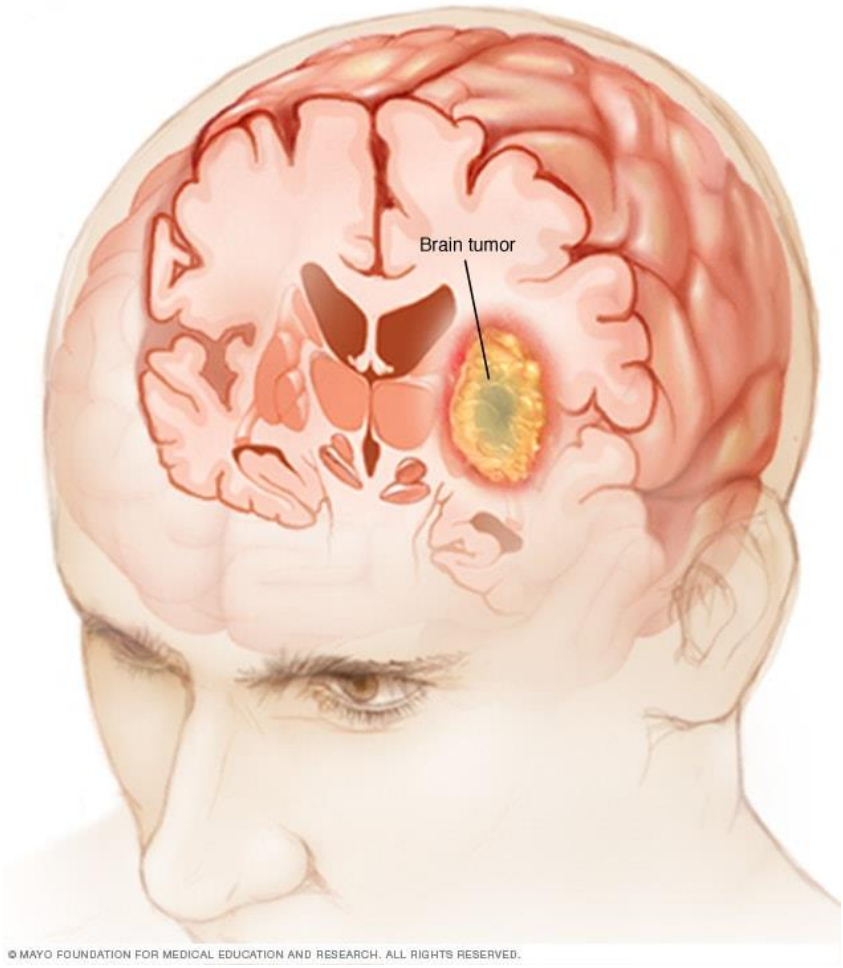


We have lost only 6 cases of Pneumonia however with a cost in accuracy which is not the goal for this kind of task, since we pay attention in Type II error (False Negative).

2. Brain Segmentation

A brain tumour is a growth of abnormal cells in the brain. The anatomy of the brain is very complex, with different parts responsible for different nervous system functions. Brain tumours can develop in any part of the brain or skull, including its protective lining, the underside of the brain (skull base), the brainstem, the sinuses and the nasal cavity, and many other areas. There are more than 120 different types of tumours that can develop in the brain, depending on what tissue they arise from. A brain tumour is a specific type of brain lesion. A lesion describes any area of damaged tissue. All tumours are lesions, but not all lesions are tumours. Other brain lesions can be caused by stroke, injury, encephalitis and arteriovenous malformation.

As it is known and obvious once more the early identification is vital. Once more the computer vision will help us identify possible brain anomalies by analysing MRI photos.



Data Set

This dataset contains brain MR images together with manual FLAIR abnormality segmentation masks. The images were obtained from The Cancer Imaging Archive (TCIA). They correspond to 110 patients included in The Cancer Genome Atlas (TCGA) lower-grade glioma collection with at least fluid-attenuated inversion recovery (FLAIR) sequence and genomic cluster data available.

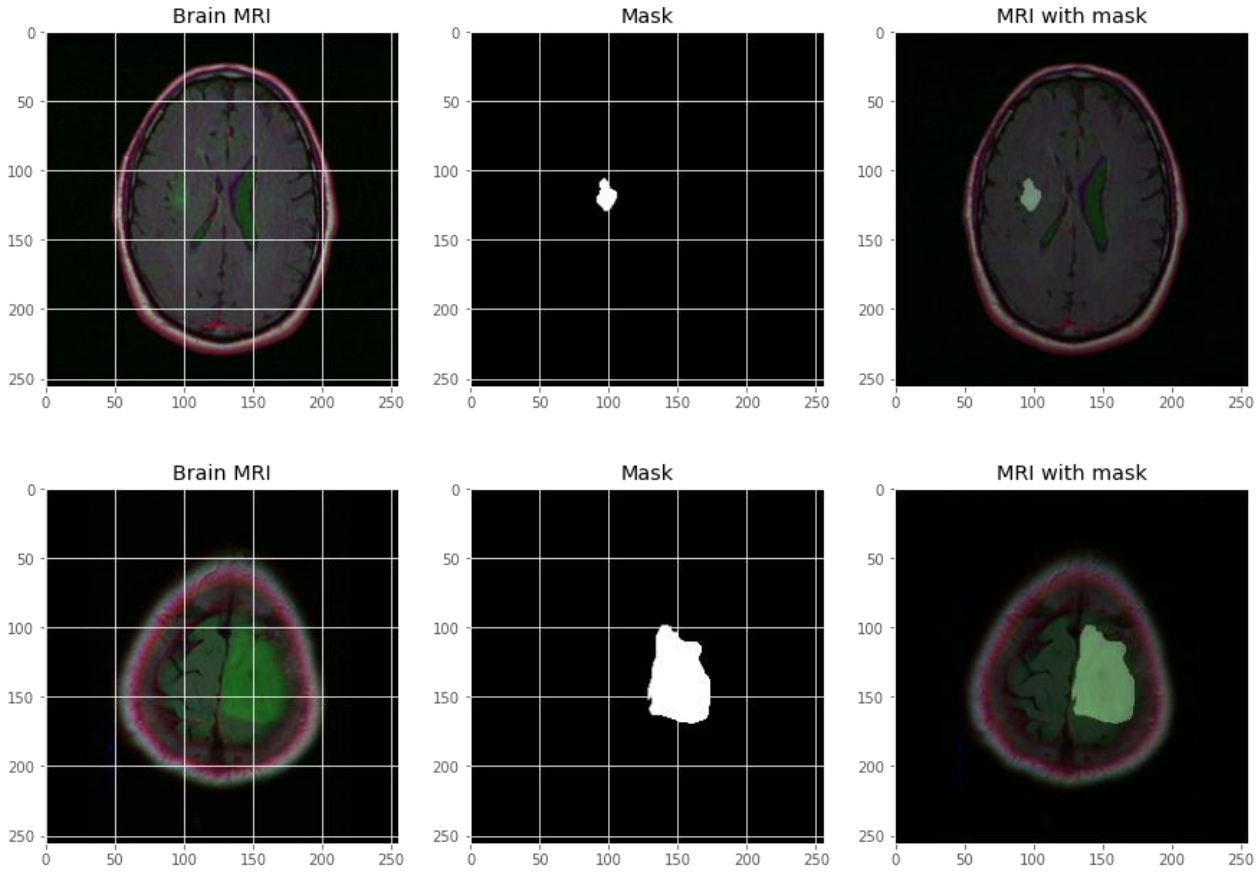
The MRI consist of several images and thus 110 patients MRI comprise of 3929 images and masks. Each image has a respective mask which depicts the area of abnormality in the MRI. The

initial data are available in cancer imaging archive[2]. The collection methodology for the segmentation of masks performed by specialized radiologists.

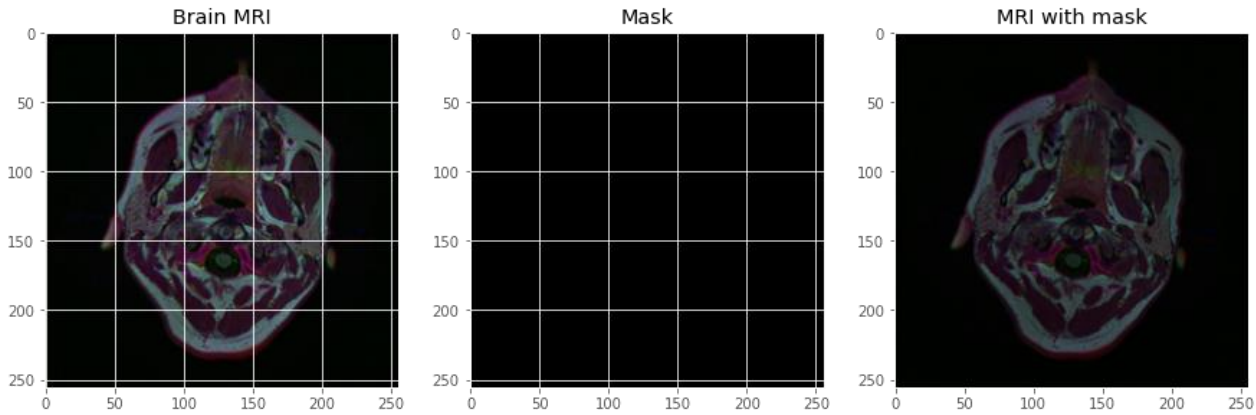
Our task is to create an algorithm which will identify the abnormality segment in the brain from the given MRI.

Preprocessing

Initially we created a dataframe with all the directories and the names of the images and the respective masks. This helped us to import them in data generators in the preprocessing phase. In order to understand better the data, we plot some MRI's with their masks. Some results are shown below:



[2] https://wiki.cancerimagingarchive.net/display/Public/TCGA-LGG



As can be seen in the figures above the masks indicate the exact position of the tumour in the same scale. It has to be mentioned that if we combine the two images, we can see the tumour clearly.

Preprocessing is a vital process before feeding the data into any algorithm. In our case the data are images and masks. Before feeding them into the algorithm the following steps were applied:

- Image Augmentation
  For the image augmentation we applied rotation, width shift, height shift, shear, zoom and horizontal flip.

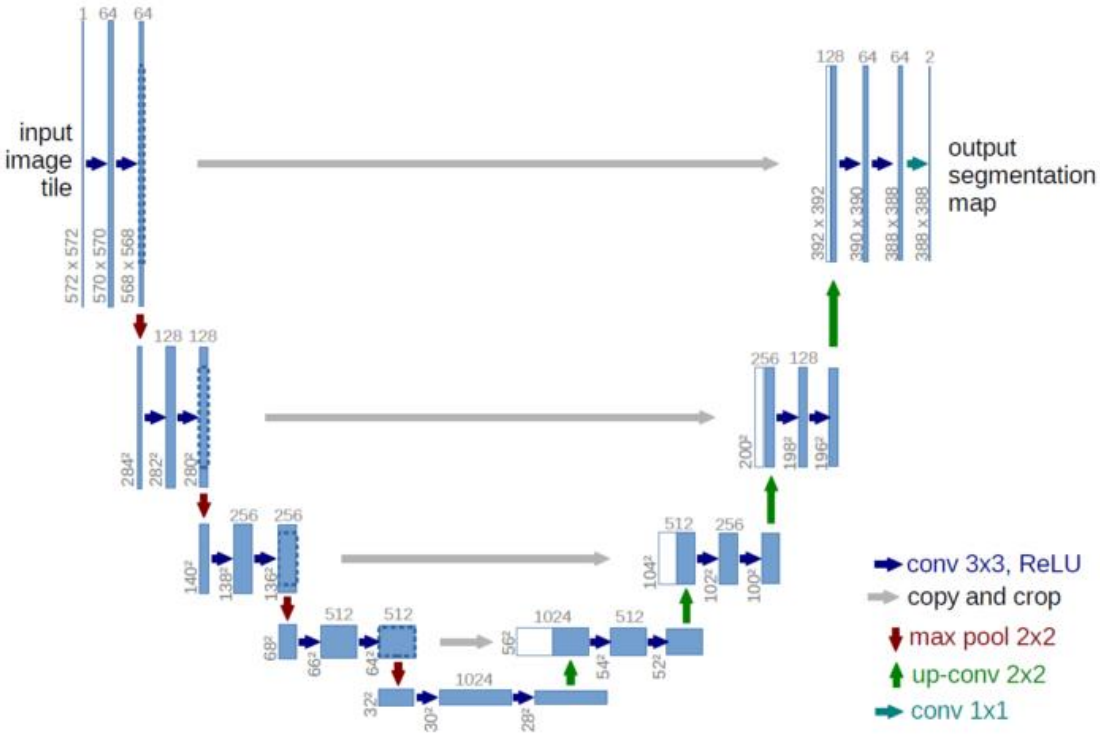
- Reshape the images
  We reshaped the images to a shape of (256, 256, 3) for MRI's and (256, 256, 1) for the masks since they are in grayscale and have only one channel.

- Normalize the images in order to help algorithm perform better
  Divide all the pixels with 255 which is the max value in order to range from 0 to 1.

- Split the images and masks in train and validation sets

Model

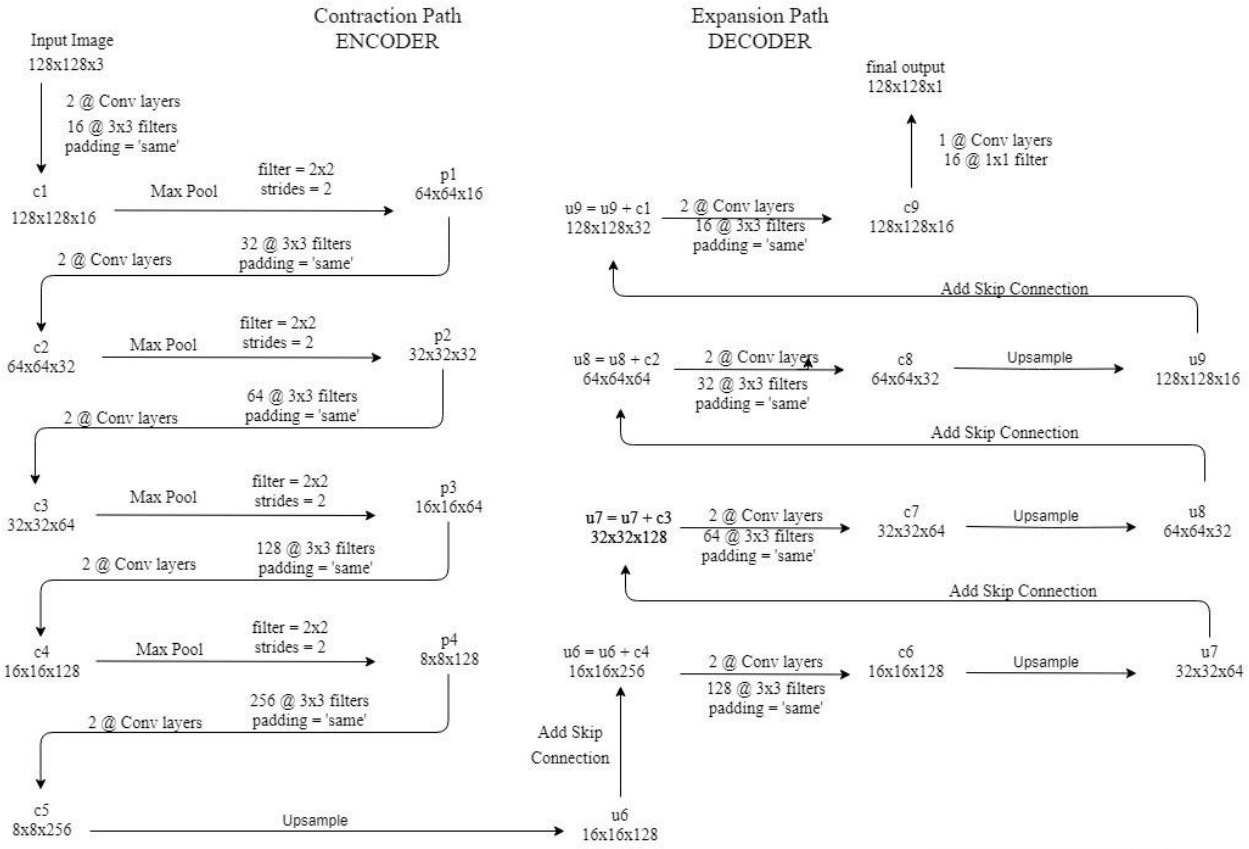
Unet was developed by Olaf Ronneberger et al[3]. for Bio Medical Image Segmentation. The architecture contains two paths. First path is the contraction path (also called as the encoder) which is used to capture the context in the image. The encoder is just a traditional stack of convolutional and max pooling layers. The second path is the symmetric expanding path (also called as the decoder) which is used to enable precise localization using transposed convolutions. Thus, it is an end-to-end fully convolutional network (FCN), which only contains Convolutional layers and does not contain any Dense layer because of which it can accept image of any size. In the original paper, the Unet architecture in depicted in the figure below:



Each blue box in the above architecture corresponds to a multi-channel feature map. The number of channels is denoted on top of the box. The analytical architecture is shown in the next figure[4]:

[3] https://arxiv.org/pdf/1505.04597.pdf

[4] The image was taken from https://towardsdatascience.com/understanding-semantic-segmentation-with-unet-6be4f42d4b47



- The left hand side is the contraction path (Encoder) where we apply regular convolutions and max pooling layers.
- In the Encoder, the size of the image gradually reduces while the depth gradually increases. Starting from 128x128x3 to 8x8x256
- The right hand side is the expansion path (Decoder) where we apply transposed convolutions along with regular convolutions
- In the decoder, the size of the image gradually increases and the depth gradually decreases. Starting from 8x8x256 to 128x128x1
- Intuitively, the Decoder recovers the information (precise localization) by gradually applying up-sampling
- To get better precise locations, at every step of the decoder we use skip connections by concatenating the output of the transposed convolution layers with the feature maps from the Encoder at the same level:

  u6 = u6 + c4

u7 = u7 + c3

u8 = u8 + c2

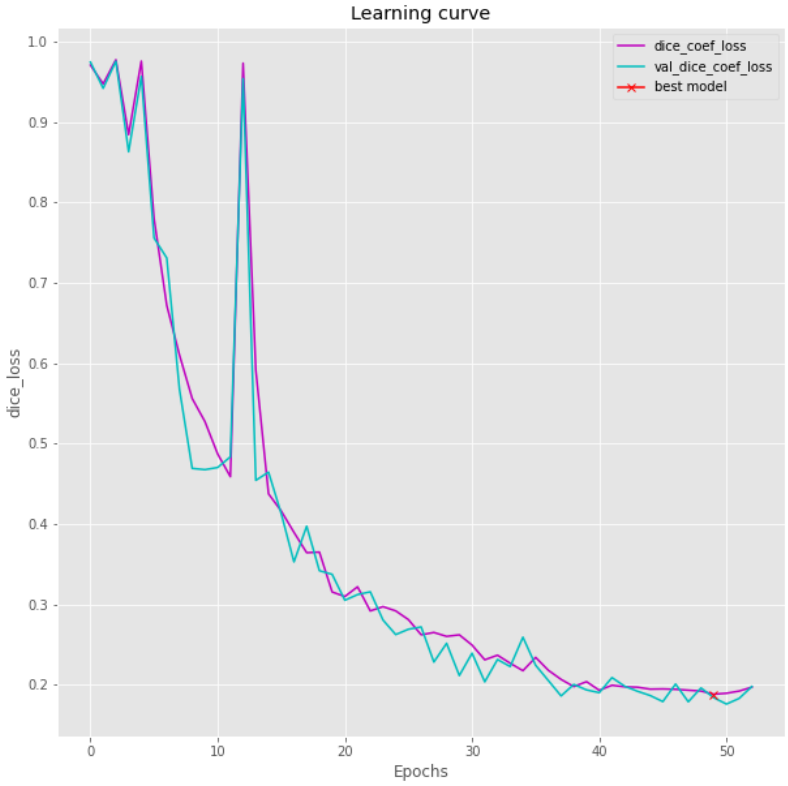
u9 = u9 + c1

After every concatenation we again apply two consecutive regular convolutions so that the model can learn to assemble a more precise output

- This is what gives the architecture a symmetric U-shape, hence the name UNET

Unet can be performed with sophisticated deep convolution neural networks like deep resnets and densenets. We applied a simple architecture of Unet with 4 convolution layers and respective transpose convolution layers. It has to be mentioned that we used batch normalization in several steps in order to facilitate the training and Dropouts in order to avoid overfitting.

The training of the model performed for 100 epochs and we used during the training early stopping with patience of 10 epochs, learning rate update with patience of 5 epochs and we save the best model during training. The Learning curve is depicted in the figure below:



Results

In order to assess the model we used Dice Loss which originates from Sorensen-Dice coefficient statistic developed in 1940's to gauge the similarity between two samples and was brought to computer vision community by Milletari et al.[5] in 2016 for medical image segmentation. The dice coefficient is calculated with the following function:

$$D = \frac{2\sum_i^N p_i g_i}{\sum_i^N p_i^2 + \sum_i^N g_i^2}$$

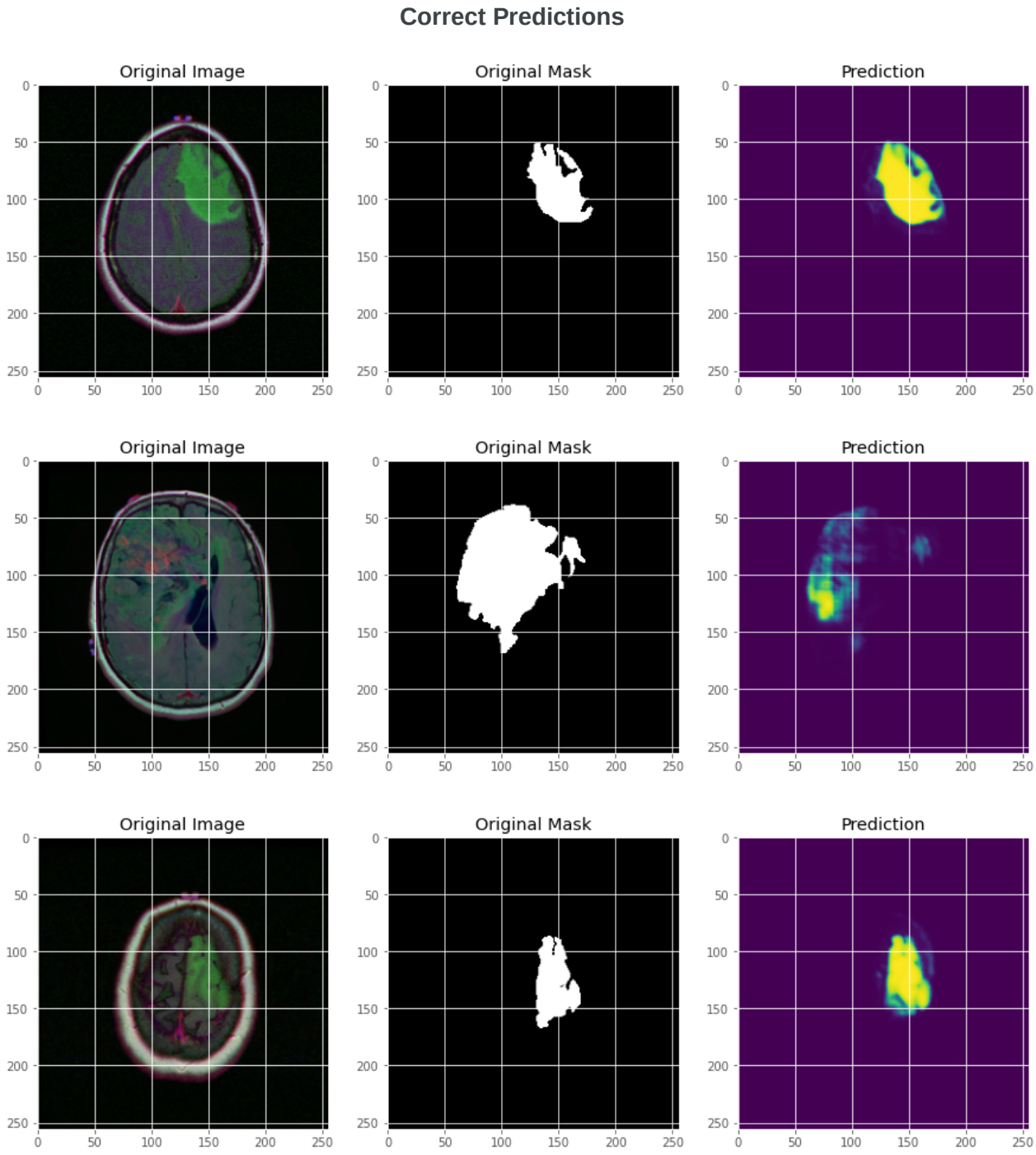
The above equation depicts the Dice coefficient, in which pi and gi represent pairs of corresponding pixel values of prediction and ground truth, respectively. In boundary detection scenario, the values of pi and gi are either 0 or 1, representing whether the pixel is boundary (value of 1) or not (value of 0). Therefore, the denominator is the sum of total boundary pixels of both prediction and ground truth, and the numerator is the sum of correctly predicted boundary pixels because the sum increments only when pi and gi match. In plain words it compares the predicted boundaries pixels with the ground truth.

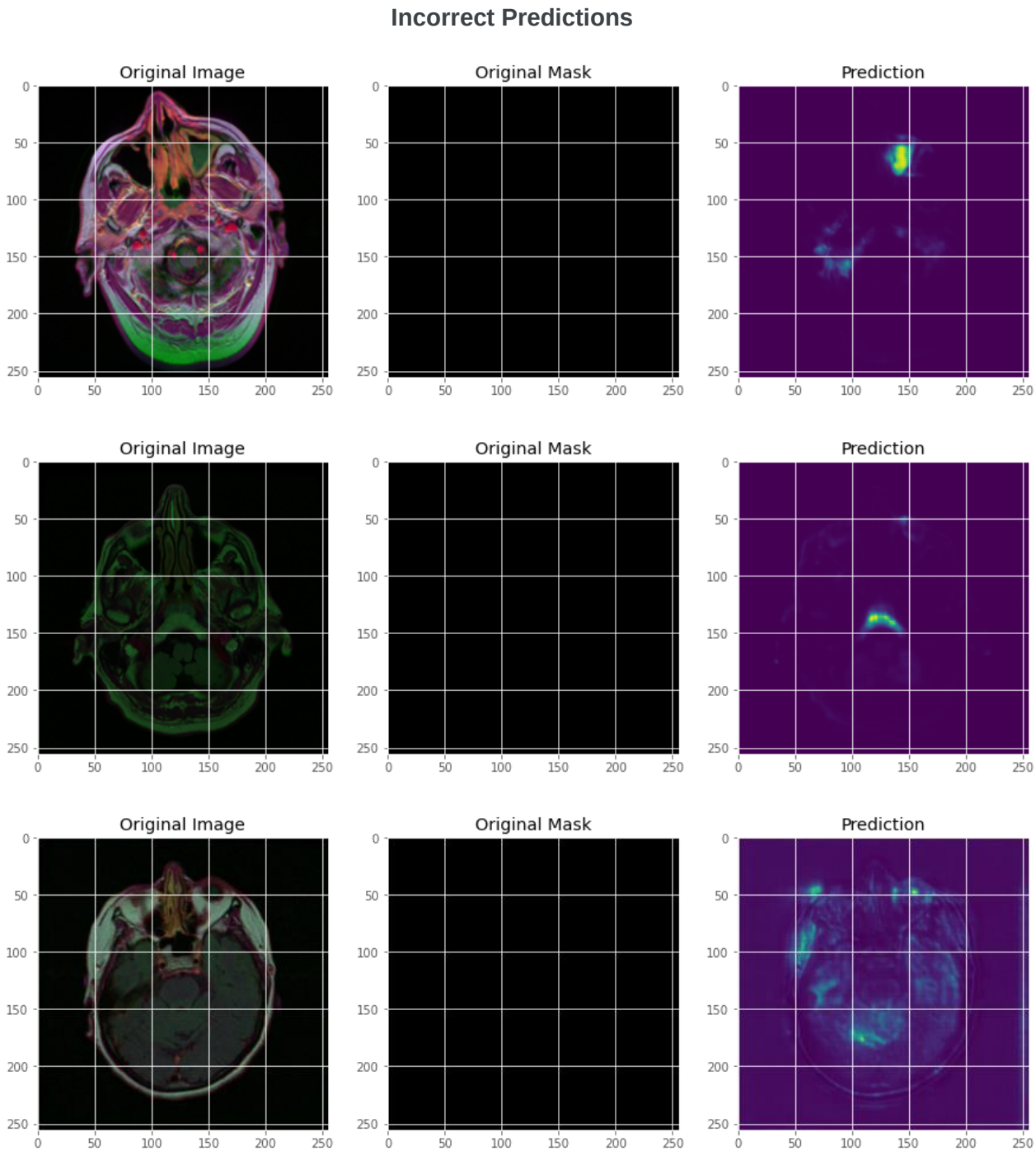
We performed two trials with different loss functions in the compiler: binary cross entropy and dice coefficient loss:

- For binary cross entropy the dice coefficient equals to 0.8306.
- For dice coefficient loss the loss the dice coefficient equals to 0.8532.

[5] F. Milletari, N. Navab and S. Ahmadi, "V-Net: Fully Convolutional Neural Networks for Volumetric Medical Image Segmentation," *2016 Fourth International Conference on 3D Vision (3DV)*, 2016, pp. 565-571, doi: 10.1109/3DV.2016.79.

The best way to see results is to depict the predicted masks against the original masks:





We can see that the results are accurate to identify the mask however there are some mistakes as it was expected. Further hyperparameter tuning and a deeper encoder in the Unet will significantly increase its performance however it was out of our computational capacity.