BRAIN TUMOUR SEGMENTATION

Under the Supervision of Dr J.S. Sahambi

Department of Electrical Engineering

Indian Institute of Technology Ropar

Students:

Nikhil Nischal 2018eeb1169
Preetesh Verma 2018eeb1171

Problem Statement

Brain tumour segmentation is a process of identifying the cancerous brain tissues and labelling them automatically based on the tumour types. Manual segmentation of tumours from brain MRI is time-consuming and error-prone. There is a need for a fast and accurate brain tumour segmentation technique.

We aim to develop a deep learning model using a U-Net with adaptations in the training and testing strategies, network structures, and model parameters for brain tumour segmentation.

About Data

This dataset contains brain MR images together with manual FLAIR abnormality segmentation masks.

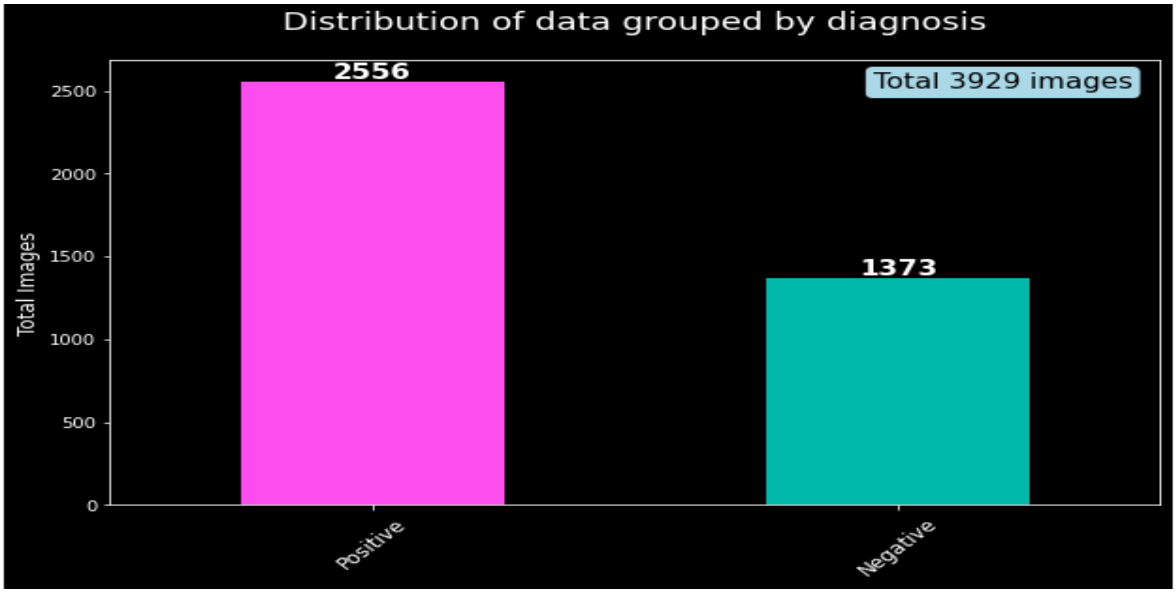
The images were obtained from The Cancer Imaging Archive (TCIA).

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.

Tumour genomic clusters and patient data is provided in the data.csv file.

The dataset is made public on Kaggle and contains a directory structure of 110 directories with each directory containing several files of images and their corresponding masks(minimum 20 files and maximum of 84 files).

The following is the visualization of the same with the number of Positive files for Tumour being 2556 and Non-Tumour being 1373 bringing a total of 3929 images.



The following image is a random sequence of 5 images with the images and their masks and the combination of the two.

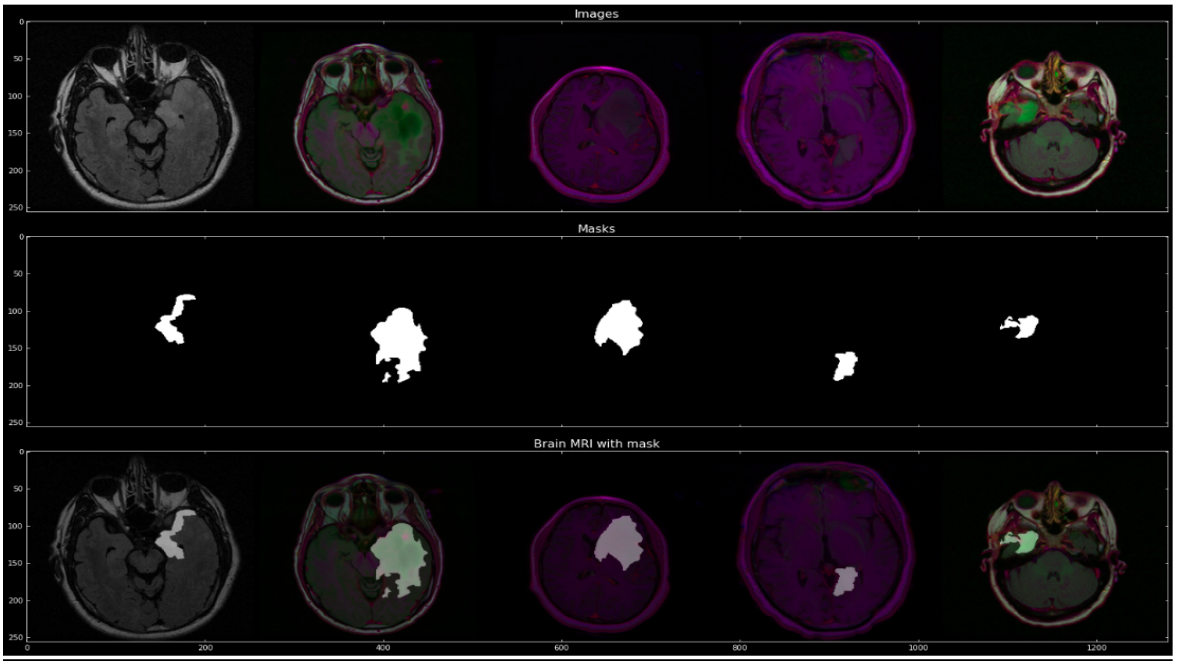


Figure 14: Brain tumour images

The following image describes the patient wise distribution of positive and negative images for tumours. The patient ID is on the horizontal axis and the number of images corresponds to the patient on the vertical.

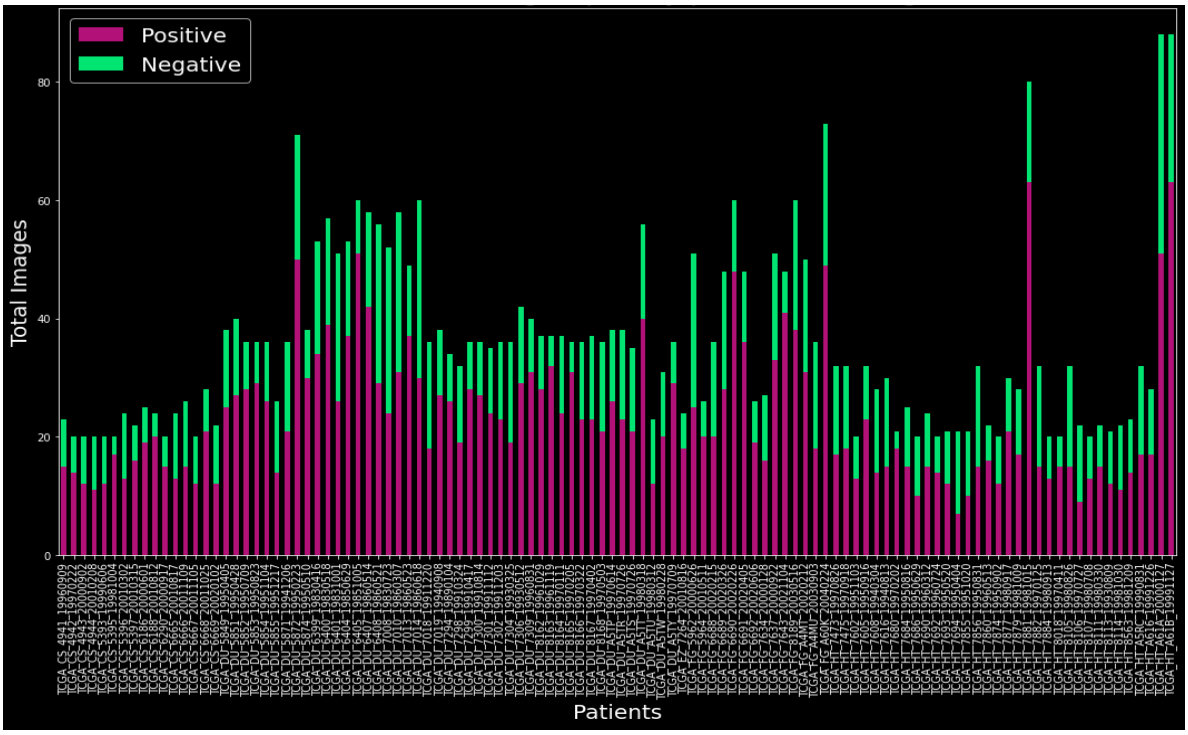


Figure 15: Distribution of data grouped by patients and diagnosis

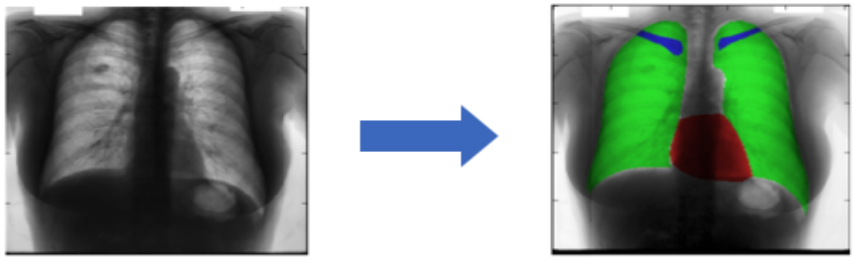
Model Description

Github Link—Code
Model Information

Model Name	No. Of Epochs	Metric used	Loss Function	Paper Link
UNet	100	Binary Accuracy, IOU	Dice Coeff	Link
UNet ++	50	Binary Accuracy, IOU	Dice Coeff	Link
ResUNet	100	Binary Accuracy, IOU	Dice Coeff	Link
Attention UNet	100	Binary Accuracy, IOU	Cross-Entropy	Link
K Folded UNet	40	Binary Accuracy, IOU	Dice Coeff	—----

Image Segmentation

One of the crucial requirements for the successful training of deep neural networks is that they require thousands of annotated training samples.
In the last two years, deep convolutional networks have outperformed the state of the art in many visual recognition tasks. The typical use of convolutional networks is on classification tasks, where the output to an image is a single class label. However, in many visual tasks, especially in biomedical image processing, the desired output should include localization, i.e., a class label is supposed to be assigned to each pixel.
The goal of semantic image segmentation is to label each pixel of an image with a corresponding class of what is being represented. Because we're predicting for every pixel in the image, this task is commonly referred to as dense prediction. The expected output in semantic segmentation are not just labels and bounding box parameters. The output itself is a high-resolution image (typically of the same size as the input image) in which each pixel is classified to a particular class. Thus it is pixel-level image classification.



Input Image

Segmented Image

Some Basic Operations

i. Convolution operation

There are two inputs to convolutional operation

i) A 3D volume (input image) of size (nin x nin x channels)

ii) A set of 'k' filters (also called kernels or feature extractors) each one of size (f x f x channels), where f is typically 3 or 5.

The output of a convolutional operation is also a 3D volume (also called output image or feature map) of size (nout x nout x k).

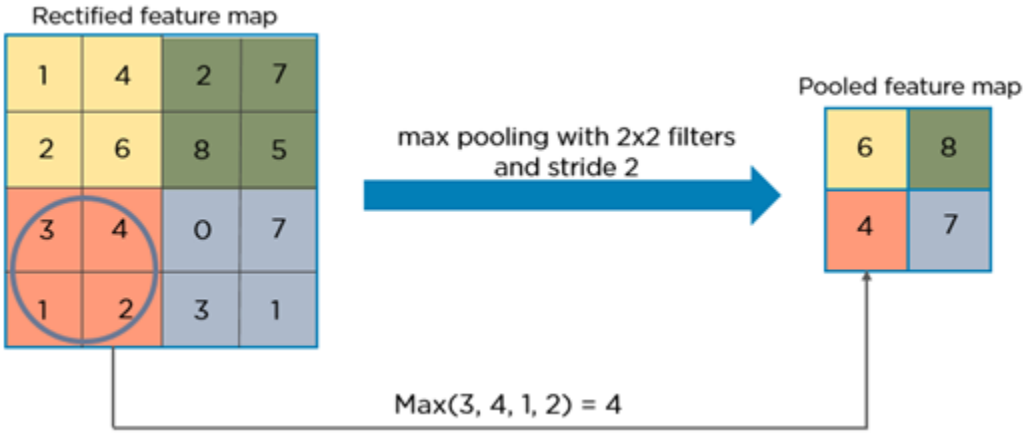
The relationship between nin and nout is as follows

$$n_{out} = \left[\frac{n_{in} + 2p - k}{s}\right] + 1$$

n_{in}: number of input features
n_{out}: number of output features
k: convolution kernel size
p: convolution padding size
s: convolution stride size

Ii. Max Pooling Operation

In simple words, the function of pooling is to reduce the size of the feature map so that we have fewer parameters in the network. The idea is to retain only the important features (max valued pixels) from each region and throw away the information which is not important. By important, I mean that information best describes the context of the image.



Iii. Transposed Convolution

The transposed convolution operation forms the same connectivity as the normal convolution but in the backward direction.

We can use it to conduct up-sampling. Moreover, the weights in the transposed convolution are learnable. So we do not need a predefined interpolation method.

Even though it is called the transposed convolution, it does not mean that we take some existing convolution matrix and use the transposed version. The main point is that the association between the input and the output is handled in a backward fashion compared with a standard convolution matrix (one-to-many rather than many-to-one association).

Summary

- Convolution and pooling operations downsample the image, i.e. convert a high-resolution image to a low-resolution image
- Max Pooling operation helps to understand “WHAT” is there in the image by increasing the receptive field. However, it tends to lose the information of “WHERE” the objects are.
- In semantic segmentation, it is not just important to know “WHAT” is present in the image but it is equally important to know “WHERE” it is present. Hence we need a way to upsample the image from low resolution to high resolution which will help us restore the “WHERE” information.
- Transposed Convolution is the most preferred choice to perform upsampling, which learns parameters through backpropagation to convert a low-resolution image to a high-resolution image.

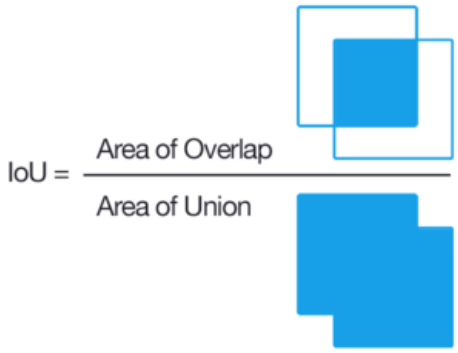
Metrics:

Binary Pixel accuracy

It is the percent of pixels in your image that are classified correctly.class imbalance. When our classes are extremely imbalanced, it means that a class or some classes dominate the image, while some other classes make up only a small portion of the image.

IOU

The Intersection-Over-Union (IoU), also known as the Jaccard Index, is one of the most commonly used metrics in semantic segmentation… and for good reason. The IoU is an extremely effective and very straightforward metric.

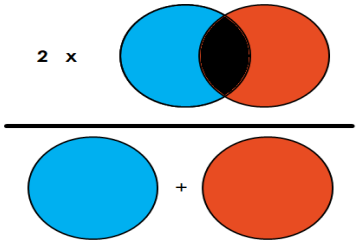


Before reading the following statement, take a look at the image to the left. Simply put, the IoU is the area of overlap between the predicted segmentation and the ground truth divided by the area of union between the predicted segmentation and the ground truth, as shown on the image to the left. This metric ranges from 0–1 (0–100%) with 0 signifying no overlap and 1 signifying perfectly overlapping segmentation.
For binary (two classes) or multi-class segmentation, the mean IoU of the image is calculated by taking the IoU of each class and averaging them. (It's implemented slightly differently in code).

Loss Function

Dice Coefficient

Simply put, the Dice Coefficient is 2 * the Area of Overlap divided by the total number of pixels in both images. (See explanation of the area of the union in section 2).



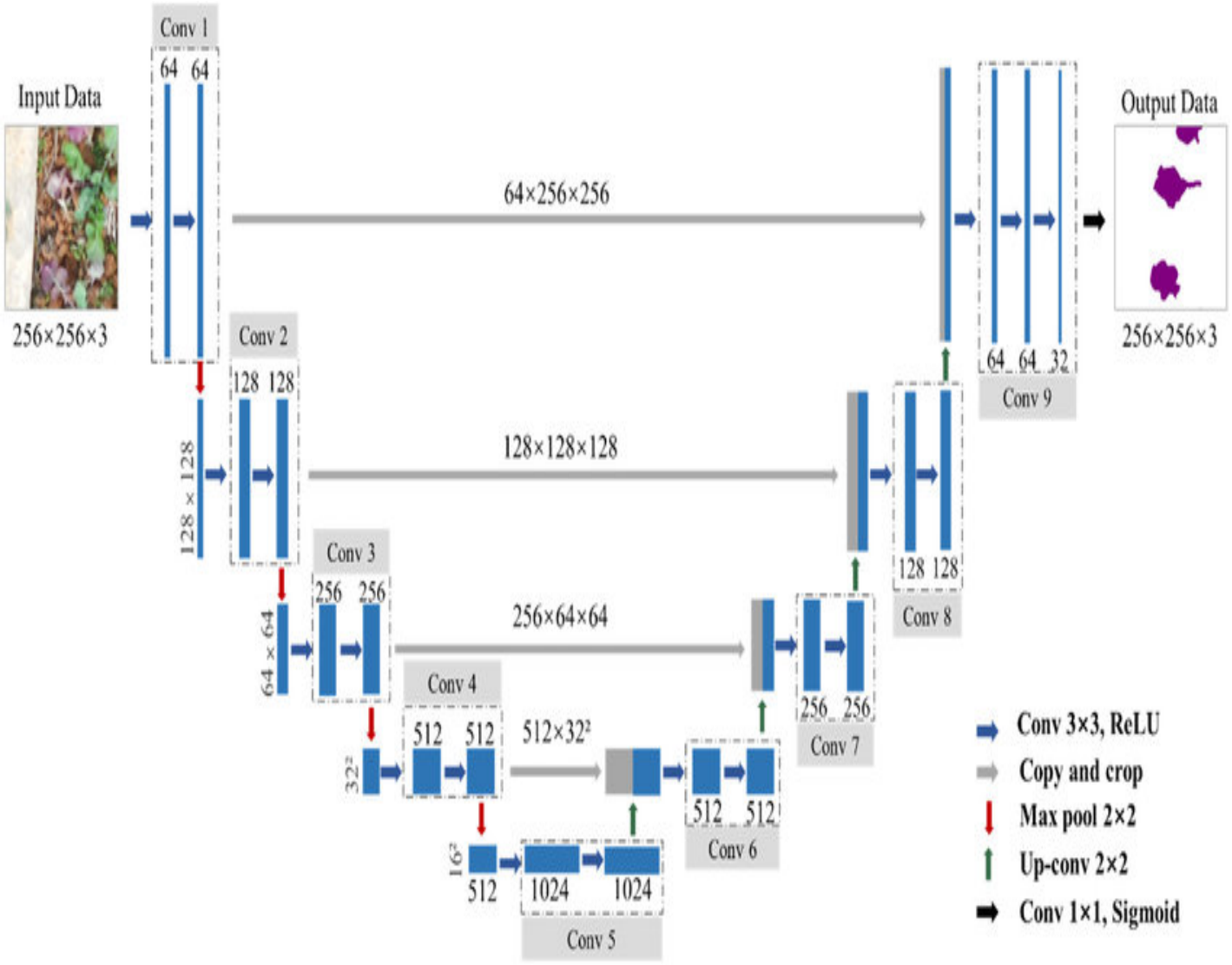
UNet Model

UNet is a network whose architecture consists of a contracting path to capture context and a symmetric expanding path that enables precise localization.
The U-Net model has shown the good performance of semantic segmentation, especially in biomedical image segmentation. U-Net used in this study has a typical encoder-decoder architecture.
The main idea in the "fully convolutional network" is to supplement a usual contracting network by successive layers along with pooling operations and then an expansive network, where pooling operators are replaced by upsampling operators. Hence, these

layers increase the resolution of the output. As a consequence, the expansive path is more or less symmetric to the contracting path and yields a u-shaped architecture.

Network Architecture

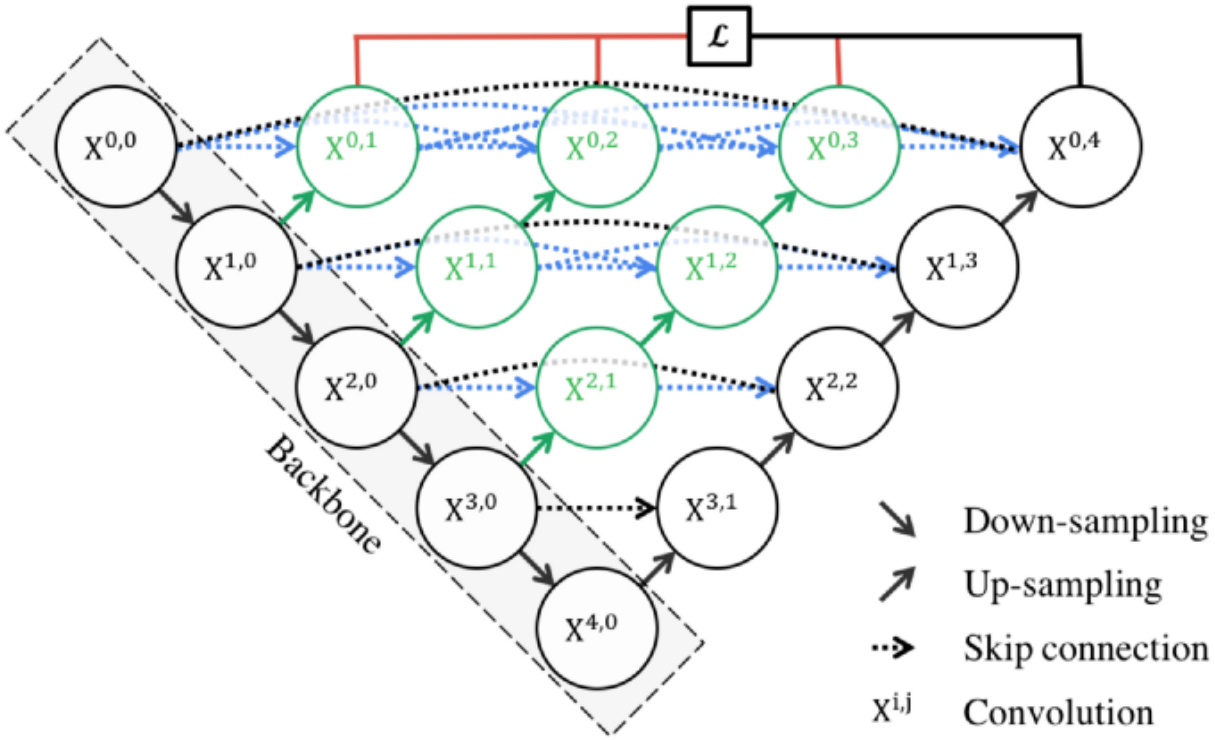
It consists of a contracting path (left side) and an expansive path (right side). The contracting path follows the typical architecture of a convolutional network.
The part of the encoder consists of the repeated convolutional layer with a kernel size of 3 × 3 each followed by a ReLU and each followed by a 2 × 2 max-pooling with a stride of 2, which gradually reduces the patch sizes. The part of the decoder contains the up-sampling operations by a 2 × 2 convolution, which halves the number of feature channels. As the main difference of the U-Net model compared with other encoder-decoder architectures, the skip connections could concatenate feature maps in up-sampling with feature maps of the same level in the encoder, which could contribute the decoder to better recovering and optimizing the details of the object. Moreover, it could output images of the same sizes as the input due to the usage of the same padding for filters.
These skip connections from earlier layers in the network (before a downsampling operation) should provide the necessary detail to reconstruct accurate shapes for segmentation boundaries. To localize, high-resolution features, the contracting path is combined with the upsampled output. Indeed, we can recover more fine-grain detail with the addition of these skip connections.
In this study, Adam optimizer and an initial learning rate of 10-4 were used. The training batch size was set to 32 and the epoch was set to 100. For binary classification in this study, the U-Net was trained with a Dice Coefficient, which can handle the imbalanced number of pixels for each class, and a Sigmoid function as the activation layer.



Unet ++

UNet++ have 3 additions to the original U-Net:

1. redesigned skip pathways (shown in green)
2. dense skip connections (shown in blue)
3. deep supervision (shown in red)



Redesigned skip pathways

In UNet++, the redesigned skip pathways (shown in green) have been added to bridge the semantic gap between the encoder and decoder subpaths.

The purpose of these convolutions layers is aimed at reduce the semantic gap between the feature maps of the encoder and decoder subnetworks. As a result, it is possibly a more straightforward optimisation problem for the optimiser to solve. The hypothesis is that the optimizer would face an easier optimization problem when the received encoder feature maps and the corresponding decoder feature maps are semantically similar.

Skip connections used in U-Net directly connects the feature maps between encoder and decoder, which results in fusing semantically dissimilar feature maps.

However, with UNet++, the output from the previous convolution layer of the same dense block is fused with the corresponding up-sampled output of the lower dense block. This brings the semantic level of the encoded feature closer to that of the feature maps waiting in the decoder; thus optimisation is easier when semantically similar feature maps are received.

All convolutional layers on the skip pathway use kernels of size 3×3.

Dense skip connections

In UNet++, Dense skip connections (shown in blue) has implemented skip pathways between the encoder and decoder. These Dense blocks are inspired by DenseNet with the purpose to improve segmentation accuracy and improving gradient flow.
Dense skip connections ensure that all prior feature maps are accumulated and arrive at the current node because of the dense convolution block along each skip pathway. This generates full resolution feature maps at multiple semantic levels.

Deep supervision

In UNet++, deep supervision (shown in red) are added, so that model can be pruned to adjust the model complexity, to balance between speed (inference time) and performance.
For accurate mode, the output from all segmentation branches is averaged.
For fast mode, the final segmentation map is selected from one of the segmentation branches.
From the metrics table, UNet++ has outperformed U-Net in Intersection over Union but falls short in Dice coefficient.

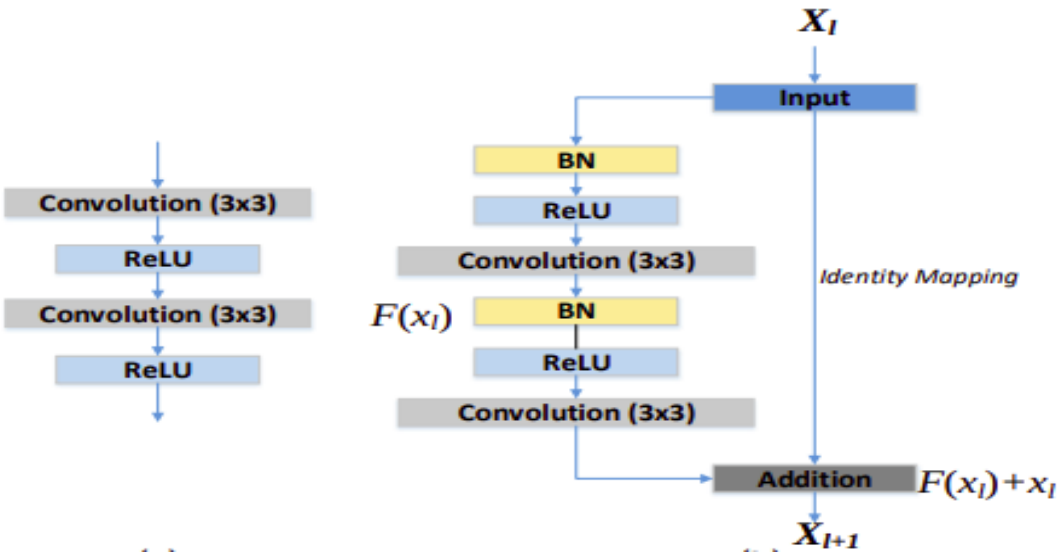
Conclusion

UNet++ aims to improve segmentation accuracy, with a series of nested, dense skip pathways.
Redesigned skip pathways made optimization easier with the semantically similar feature maps.
Dense skip connections improve segmentation accuracy and improve gradient flow.
Deep supervision allows for model complexity tuning to balance between speed and performance optimisation.

ResUNET

ResNet, a semantic segmentation model inspired by deep residual learning and UNet.
An architecture that takes advantage of both(Residual and UNet) models.
the deep residual U-Net, an architecture that takes advantage of strengths from both deep residual learning and U-Net architecture. The proposed deep residual U-Net (ResUnet) is built based on the architecture of U-Net.
This combination brings us two benefits:
1) the residual unit will ease the training of the network;
2) the skip connections within a residual unit and between low levels and high levels of the network will facilitate information propagation without degradation, making it possible to design a neural network with much fewer parameters however could achieve comparable ever better performance on semantic segmentation.
Going deeper would improve the performance of a multi-layer neural network, however, could hamper the training, and a degradation problem maybe occur. To overcome these

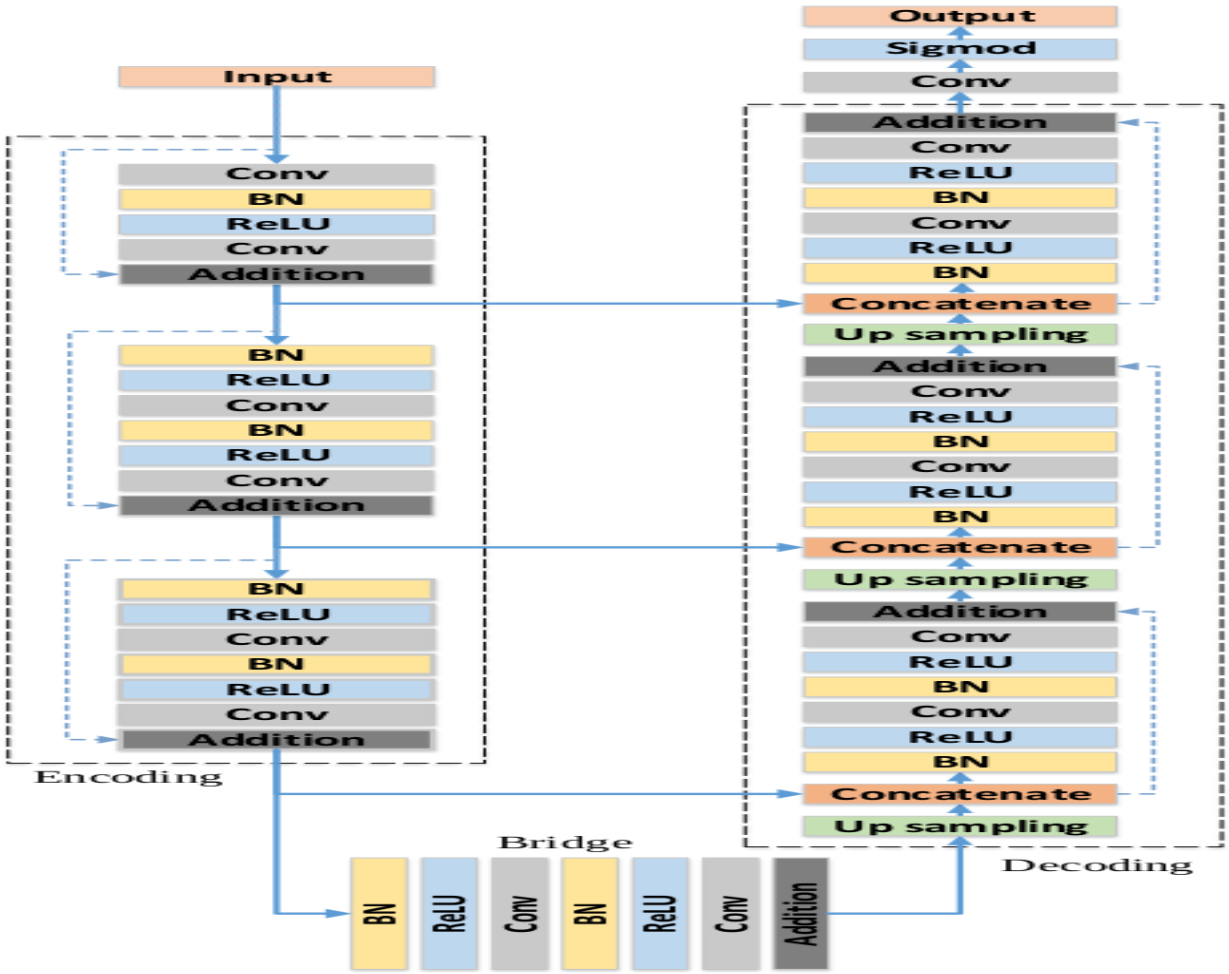
problems, proposed the residual neural network to facilitate training and address the degradation problem.



Building blocks of neural networks. (a) The plain neural unit used in U-Net and (b) residual unit with identity mapping used in the proposed ResUnet

The network comprises of three parts: encoding, bridge and decoding.1 The first part encodes the input image into compact representations. The last part recovers the representations to a pixel-wise categorization, i.e. semantic segmentation. The middle part serves as a bridge connecting the encoding and decoding paths. All of the three parts are built with residual units which consist of two 3 × 3 convolution blocks and an identity mapping. Each convolution block includes a BN layer, a ReLU activation layer and a convolutional layer. The identity mapping connects the input and output of the unit.

There are two main reasons to add skip connections: to avoid the problem of vanishing gradients, and to mitigate the Degradation (accuracy saturation) problem; where adding more layers to a suitably deep model leads to higher training error. During training, the weights adapt to mute the upstream layer, and amplify the previously-skipped layer.



Attention Unet

1. What is attention?

Attention, in the context of image segmentation, is a way to highlight only the relevant activations during training. This reduces the computational resources wasted on irrelevant activations, providing the network with better generalisation power. Essentially, the network can pay “attention” to certain parts of the image.

a. Hard Attention

Attention comes in two forms, hard and soft. Hard attention works based on highlighting relevant regions by cropping the image or iterative region proposal. Since hard attention can only choose one region of an image at a time, it has two implications, it is non-differentiable and requires reinforcement learning to train.

Since it is non-differentiable, it means that for a given region in an image, the network can either pay “attention” or not, with no in-between. As a result, standard backpropagation cannot be done, and Monte Carlo sampling is needed to calculate the accuracy across various stages of backpropagation. Considering the accuracy is

subject to how well the sampling is done, there is a need for other techniques such as reinforcement learning to make the model effective.

b. Soft Attention

Soft attention works by weighting different parts of the image. Areas of high relevance are multiplied with a larger weight and areas of low relevance is tagged with smaller weights. As the model is trained, more focus is given to the regions with higher weights. Unlike hard attention, these weights can be applied to many patches in the image. Due to the deterministic nature of soft attention, it remains differentiable and can be trained with standard backpropagation. As the model is trained, the weighting is also trained such that the model gets better at deciding which parts to pay attention to.

Hard Attention:

1. Only picks a patch at a time to pay attention to
2. Non-differentiable and requires reinforcement learning

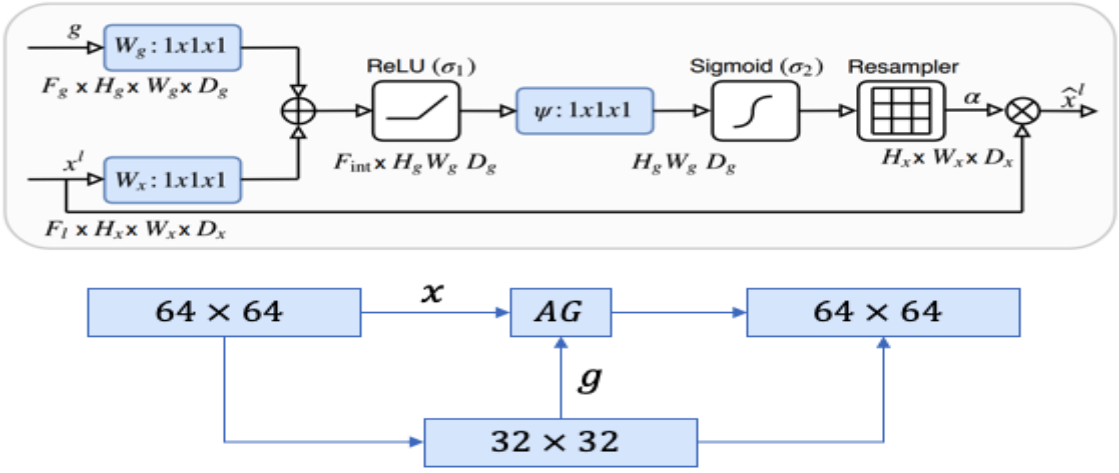
Soft Attention:

1. Weights are placed on different patches of the image to determine relevance
2. Differentiable and can be trained with backpropagation

Despite U-Net excellent representation capability, it relies on multi-stage cascaded convolutional neural networks to work. These cascaded frameworks extract the region of interest and make dense predictions. During upsampling in the expanding path, spatial information recreated is imprecise. To counteract this problem, the U-Net uses skip connections that combine spatial information from the downsampling path with the upsampling path. However, this brings across many redundant low-level feature extractions, as feature representation is poor in the initial layers. Soft attention implemented at the skip connections will actively suppress activations in irrelevant regions, reducing the number of redundant features brought across.

The attention gates implemented uses additive soft attention.

a. Breakdown of attention gates



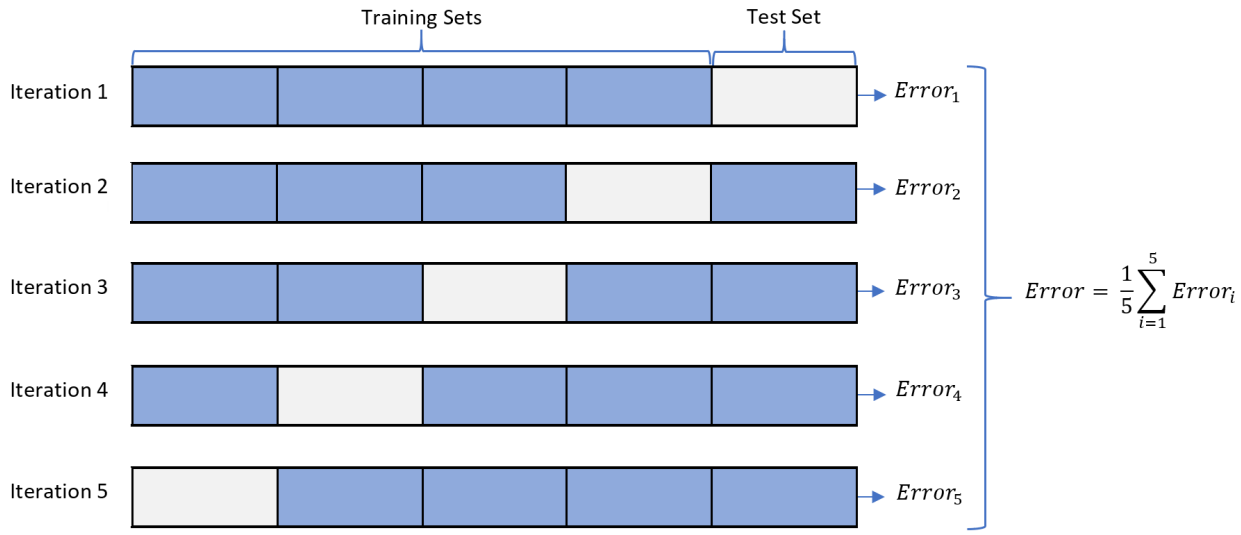
Top: Attention gate (AG) schematic. Bottom: How AGs are implemented at every skip connection.

1. The attention gate takes in two inputs, vectors x and g.
2. The vector, g, is taken from the next lowest layer of the network. The vector has smaller dimensions and better feature representation, given that it comes from deeper into the network.
3. In the example figure above, vector x would have dimensions of 64x64x64 (filters x height x width) and vector g would be 32x32x32.
4. Vector x goes through a strided convolution such that its dimensions become 64x32x32 and vector g goes through a 1x1 convolution such that its dimensions become 64x32x32.
5. The two vectors are summed element-wise. This process results in aligned weights becoming larger while unaligned weights become relatively smaller.
6. The resultant vector goes through a ReLU activation layer and a 1x1 convolution that collapses the dimensions to 1x32x32.
7. This vector goes through a sigmoid layer which scales the vector between the range [0,1], producing the attention coefficients (weights), where coefficients closer to 1 indicate more relevant features.
8. The attention coefficients are upsampled to the original dimensions (64x64) of the x vector using trilinear interpolation. The attention coefficients are multiplied element-wise to the original x vector, scaling the vector according to relevance. This is then passed along in the skip connection as normal.

We also used a grid-based gating mechanism, which takes the g vector from the upsampling path rather than the downsampling path (except for the lowest layer), as the vector would have been conditioned to spatial information from multiple scales by previous attention gates.
The differentiable nature of the attention gate allows it to be trained during backpropagation, which means the attention coefficients get better at highlighting relevant regions.

K Folded Unet

When we have limited data, dividing the dataset into Train and Validation sets may cause some data points with useful information to be excluded from the training procedure, and the model fails to learn the data distribution properly.
So, what different do we do in K-Fold cross-validation do?



K-Fold CV gives a model with less bias compared to other methods. In K-Fold CV, we have a parameter ‘k’. This parameter decides how many folds the dataset is going to be divided. Every fold gets a chance to appear in the training set (k-1) times, which in turn ensures that every observation in the dataset appears in the dataset, thus enabling the model to learn the underlying data distribution better.

Model Results

Model Name	Metric (IOU)	Loss Function (Dice Coeff)
UNet	0.8272	-0.9045
UNet ++	0.8184	-0.8378
ResUNet	0.7694	-0.8182
Attention UNet	0.3848	(binary_cross_entropy)0.0510
K Folded UNet	0.5918	-0.7680

Conclusions

Performance Of UNet

We used training set augmentation quite heavily - rotations, scaling and mirroring - and this improved classification accuracy,

Performance of UNet++

Due to enough data points already available and augmentation further increasing the data size, UNet++ did not improve much in the Metrix or loss function.

Performance of ResUNet

Since we did not have too many Convolutional Layers in the Original Network so introducing more skip connections did not improve the performance but rather led to overfitting.

Performance of K folded UNet

Cross-validation almost always lead to lower estimated errors - it uses some data that are different from the test set so it will cause overfitting for sure.

Performance of Attention UNet

Similar to the ResUNet over computation here also leads to the overfitting of the model.