Multi-Model Approach for Brain Tumor Classification and Segmentation

Deep Learning Course Exam Project
SDIC Master Degree, University of Trieste (UniTS)

October, 2024

Stefano Lusardi   Marco Tallone   Piero Zappi

Introduction

Hello, world! 😄

Hellow world in a box.

Hello world in an orange box mixed 50% with the background.

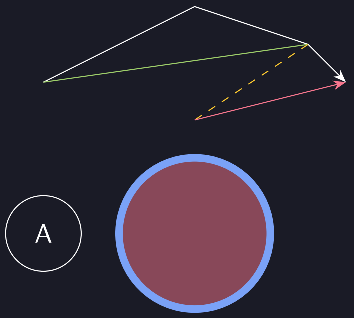
⚠️ Recall you need lualatex to compile this template.

EXAMPLE **Title of the example block**

For example like this ☻.

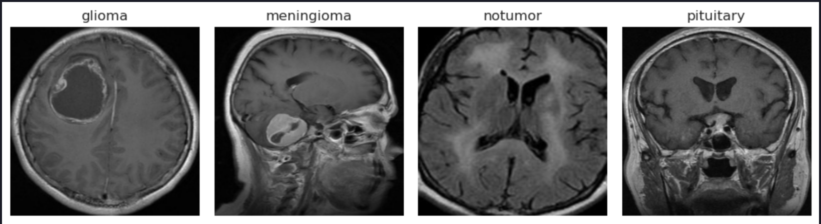
Example TikZ Picture

Minimal **working** example of a *TikZ picture*.



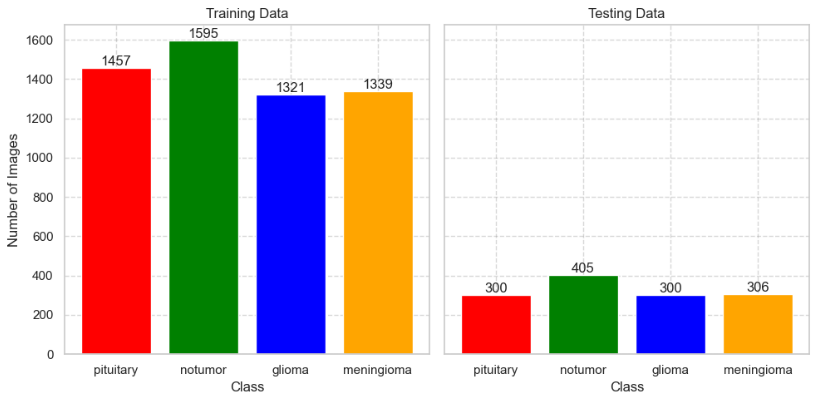
Brain Tumor MRI Dataset

- Combination of three datasets
- 7023 images of human brain MRI images
- Four classes: glioma, meningioma, no-tumor and pituitary



Brain Tumor MRI Dataset

The dataset is separated into training and testing sets with a ratio of 80% and 20% respectively



Resizing and Data Augmentation

- Images are resized to 128×128 pixels

Transformations applied to the images at each epoch:

- Random horizontal flip
- Random rotation up to 10 degrees
- Random change in brightness, contrast, saturation, and hue

Theses transformations add variability to the dataset and help the model generalize better

BraTS 2020 Dataset

- BraTS stands for Brain Tumor Segmentation
- it is composed by 155 horizontal "slices" of brain MRI images for 369 patients (volumes)

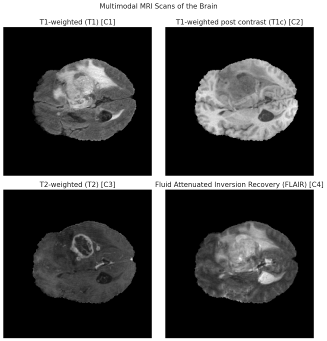
$$155 \cdot 369 = 57195$$

- we used 90% of the dataset for training and 10% for testing

BraTS 2020 Dataset

Images have 4 channels:

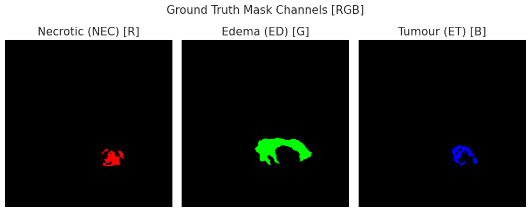
1. **T1 weighted (T1)** good for visualizing the brain but not the tumor
2. **T1 weighted with contrast (T1c)** taken with the same tecnique as T1 but with contrast
3. **T2 weighted (T2)** good for visualizing the edema
4. **Fluid Attenuated Inversion Recovery (FLAIR)** improves the visualization of the edema



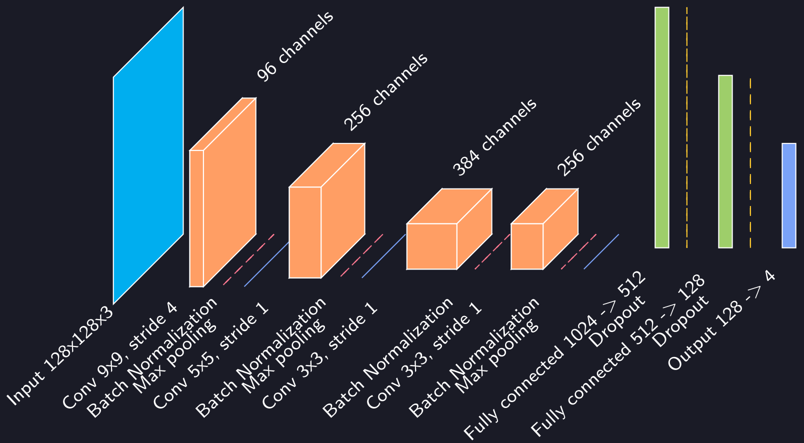
BraTS 2020 Dataset

Each slice has (eventually) 3 mask labels:

1. Necrotic and Non-Enhancing Tumour Core (NCR/NET)
2. Edema (ED)
3. Enhancing Tumour (ET)



Custom CNN Architecture



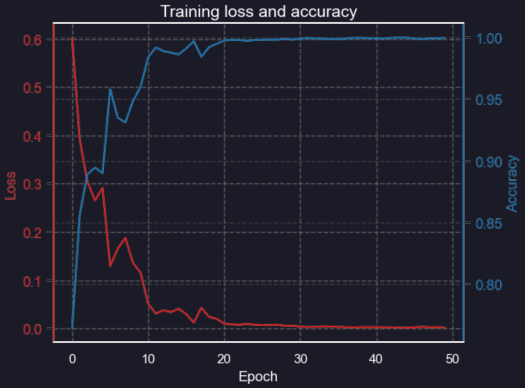
Number of parameters: 3001156

Training Details

Training parameters:

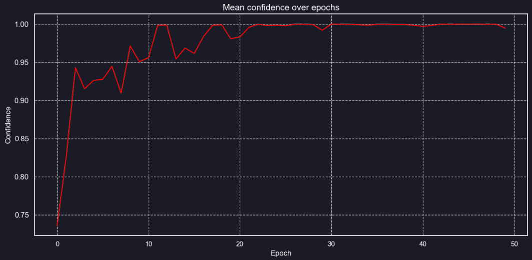
- **Epochs**: 50
- **Batch size**: 64
- **Learning rate**: 1×10^{-4}
- **Optimizer**: Adam (weight decay 1×10^{-5})
- **Scheduler**: stepLR (step size 10, gamma 0.5)
- **Loss function**: Cross-entropy
- **Activation function**: Mish
- **Dropout rate**: 0.4
- **Image size**: 128×128

Training Loss and Accuracy



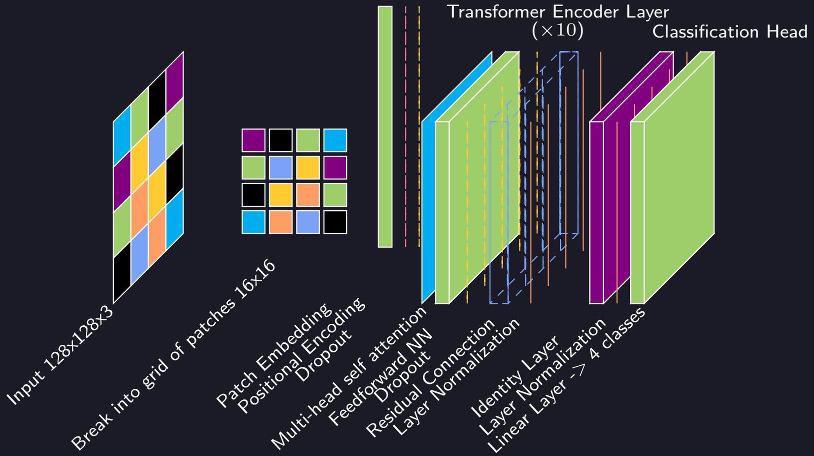
- Final training loss: 1.4×10^{-3}
- Final training accuracy: 99.9%

Confidence and Test Accuracy



- Final training confidence: 99.9%
- Final test confidence: 99.9%
- Final test accuracy: 99%

VIT Architecture



Number of parameters: 21459460

Training Details

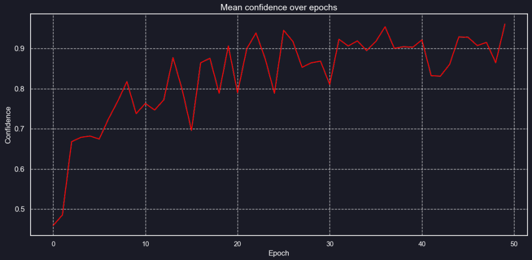
Training and model parameters:

- **Epochs**: 50
- **Batch size**: 64
- **Learning rate**: 1×10^{-4}
- **Optimizer**: Adam (weight decay 1×10^{-5})
- **Scheduler**: stepLR (step size 10, gamma 0.5)
- **Loss function**: Cross-entropy
- **Activation function**: Mish
- **Dropout rate**: 0.2
- **Image size** and **Patch size**: 128×128, 16×16
- **Number of heads**: 8
- **Number of layers**: 10
- **Patch embedding dimension**: 512
- **Feedforward dimension**: 1024

Training Loss and Accuracy

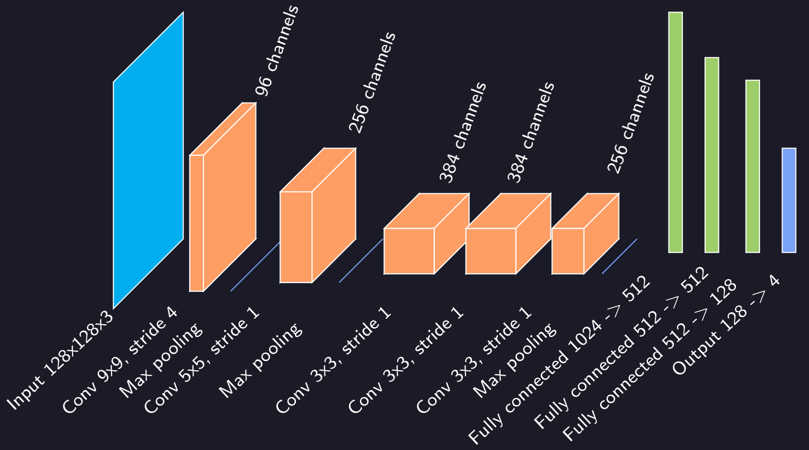
- Final training loss: 0.27
- Final training accuracy: 90%

Confidence and Test Accuracy



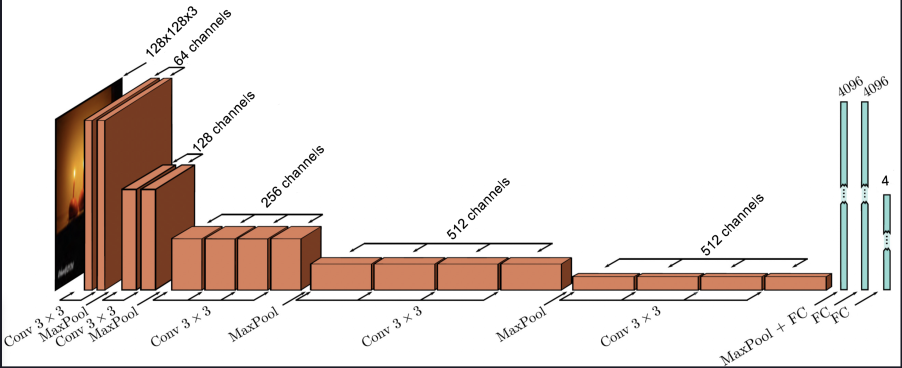
- Final training confidence: 96%
- Final test confidence: 93%
- Final test accuracy: 88%

AlexNet



Number of parameters: 4589316
Dropout rate: 0.4

VGG



Number of parameters: 4589316
Dropout rate: 0.5

Setup Differences

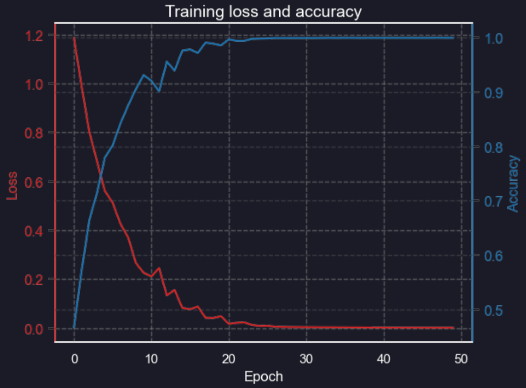
Model	Data augmentation	LR Scheduler	Activation	L2 reg.
CustomCNN	Yes	Yes	*Mish*	Yes
AlexNet	No	Yes	*ReLU*	Yes
VGG16	No	No	*ReLU*	No
VIT	Yes	Yes	*Mish*	Yes

- All the other hyperparameters and settings are the same for all models(batch size, optimizer, epochs, etc)
- Note that the **CustomCNN** is the one with less parameters $(3,001,156)$ while **VGG16** is the one with more parameters$(65,070,916)$
- **VGG16** is also the one with the highest dropout rate (0.5)

Performance Assessment

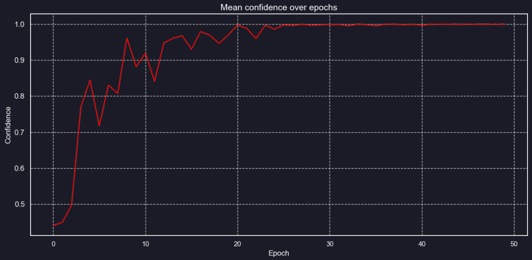
- **Loss function**: Cross-entropy loss $L(y, \hat{y}) = -\sum_i y_i \log(\hat{y}_i)$
- **Accuracy**: Number of correct predictions divided by the total number of predictions
- **Confidence**: Given by the Softmax function applied to the net output $S(x_i) = \frac{e^{x_i}}{\sum_j e^{x_j}}$

Training Loss and Accuracy for AlexNet



- Final training loss: $1.2 \cdot 10^{-3}$
- Final training accuracy: 99.9%

Confidence and Test Accuracy for AlexNet

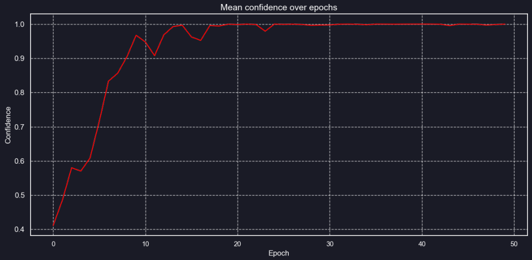


- Final training confidence: 99.9%
- Final test confidence: 96.5%
- Final test accuracy: 90%

Training Loss and Accuracy for VGG16

- Final training loss: $8.9 \cdot 10^{-6}$
- Final training accuracy: 99.9%

Confidence and Test Accuracy for VGG16



- Final training confidence: 100%
- Final test confidence: 98%
- Final test accuracy: 95%

Training Performance Comparison

Model	Loss	Accuracy	Confidence
CustomCNN	$1.4 \cdot 10^{-3}$	0.99	100%
AlexNet	$1.2 \cdot 10^{-3}$	0.99	99.9%
VGG16	$8.9 \cdot 10^{-6}$	0.99	100%
VIT	0.27	0.90	96.1%

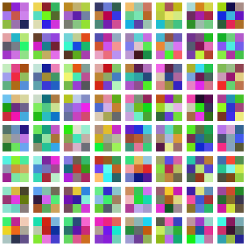
Note that these are the values reached during the last epoch.

Focus on Accuracy

Test Performance Comparison

Model	Accuracy	Confidence
CustomCNN	0.99	100%
AlexNet	0.90	96.5%
VGG16	0.95	98.0%
VIT	0.88	93.3%

Visualizing the first layer filters



U-Net Models

3 models for the segmentation task:

- **Classic U-Net**: *baseline U-Net model architecture*

U-Net Models

3 models for the segmentation task:

- **Classic U-Net**: *baseline U-Net model architecture*
- **Improved U-Net**: *small improvements, fewer parameters*

U-Net Models

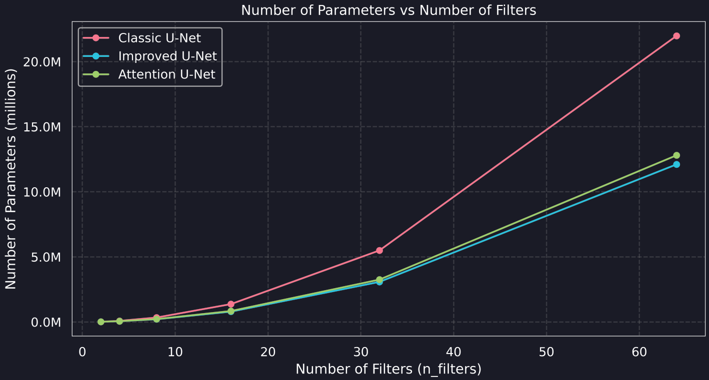
3 models for the segmentation task:

- **Classic U-Net**: *baseline U-Net model architecture*
- **Improved U-Net**: *small improvements, fewer parameters*
- **Attention U-Net**: *attention mechanism added*

U-Net Models

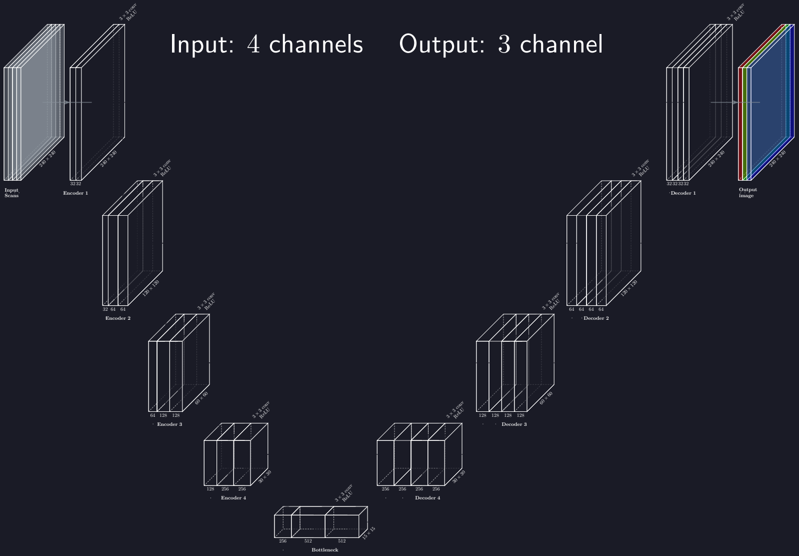
3 models for the segmentation task:

- **Classic U-Net**: *baseline U-Net model architecture*
- **Improved U-Net**: *small improvements, fewer parameters*
- **Attention U-Net**: *attention mechanism added*



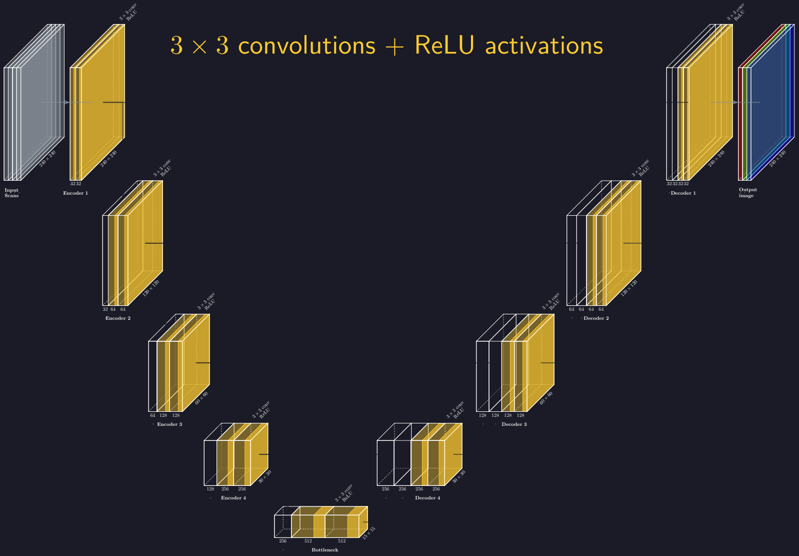
Classic U-Net

Input: 4 channels Output: 3 channel



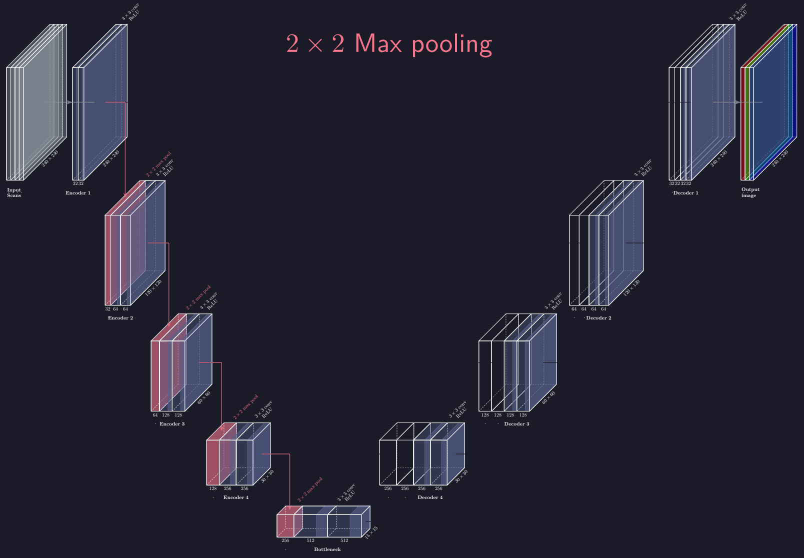
Classic U-Net

3×3 convolutions + ReLU activations



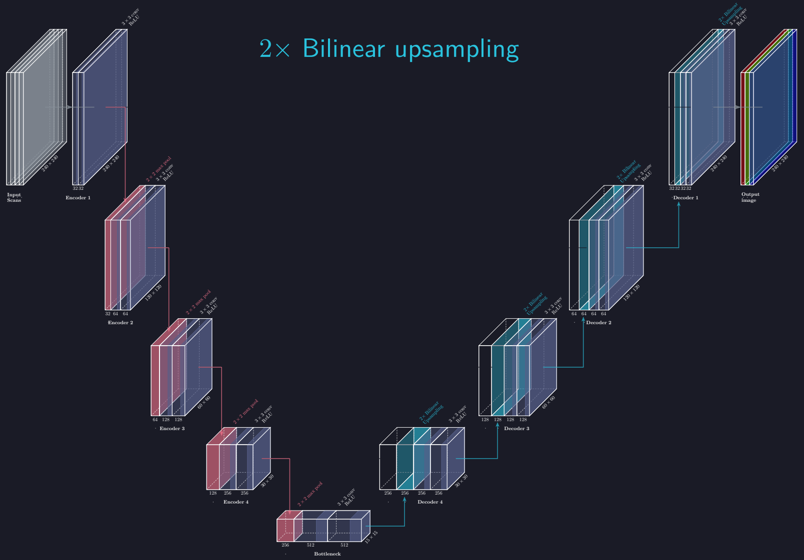
Classic U-Net

2×2 Max pooling

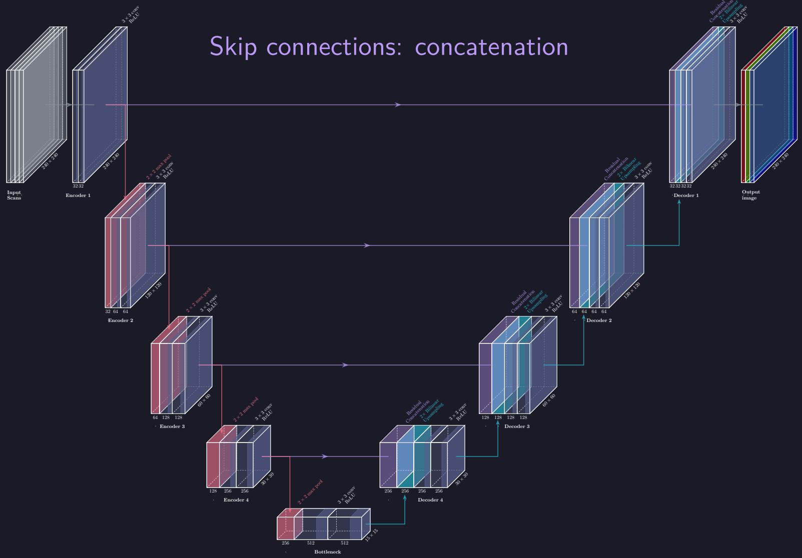


Classic U-Net

2× Bilinear upsampling



Classic U-Net

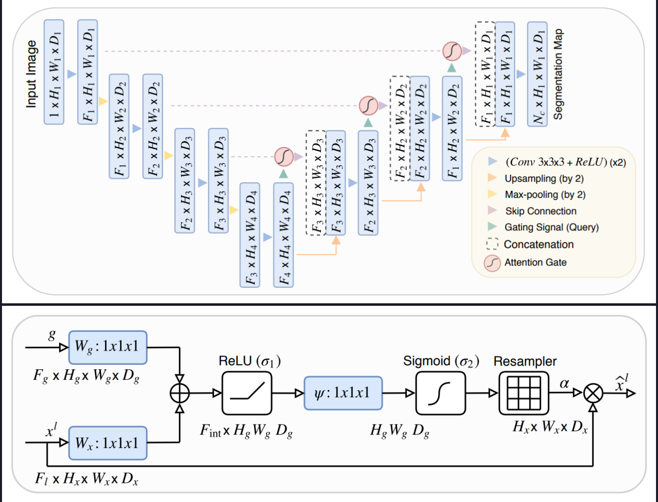


Improved U-Net

Small improvements from previous $\rightarrow$ to reduce n° of parameters and improve performance:

- **Separable Convolutions**: depthwise + pointwise convolutions
- **Batch Normalization**: to improve training and generalization
- **Larger Kernel Size**: 7×7 kernels instead of 3×3
- **Inverse Bottleneck**: expands + compresses channels
- **Additive Skip Connections**: instead of concatenated ones

Attention U-Net

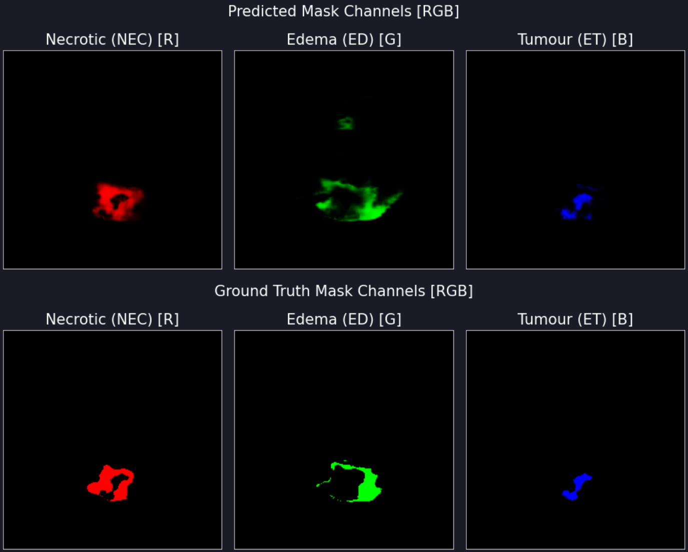


Training Details

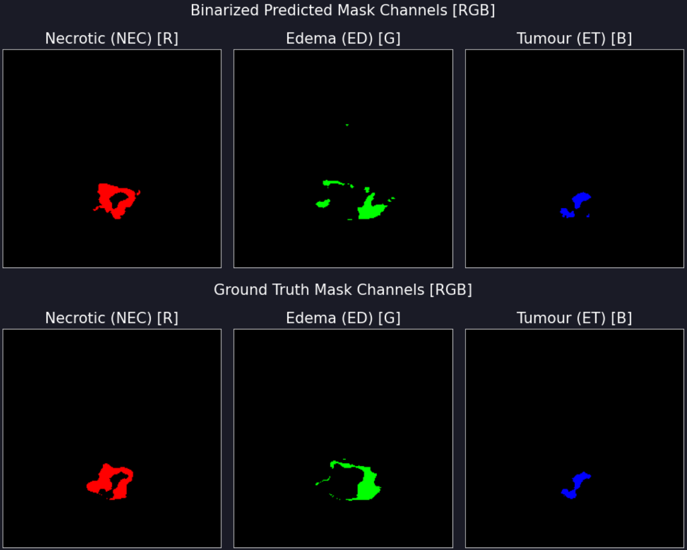
U-Net Models training parameters:

- **epochs**: 20
- **Optimizer**: Adam (with weight decay (1×10^{-2}))
- **Scheduler**: Exponential Decay $(\gamma = 0.9)$
- **Loss Function**: BCE with Logits Loss
- **learning rate**: 2×10^{-3}
- **batch size**: 32 (both training and validation)
- **image size**: 240×240
- **first encoder filters**: 32

Visualizing a prediction



Visualizing a prediction



Performance Assessment

$$\text{Dice} = \frac{2 \times |X \cap Y|}{|X| + |Y|}$$

Dice Coefficient

"overlap" metric

$$\text{Precision} = \frac{TP}{TP+FP}$$

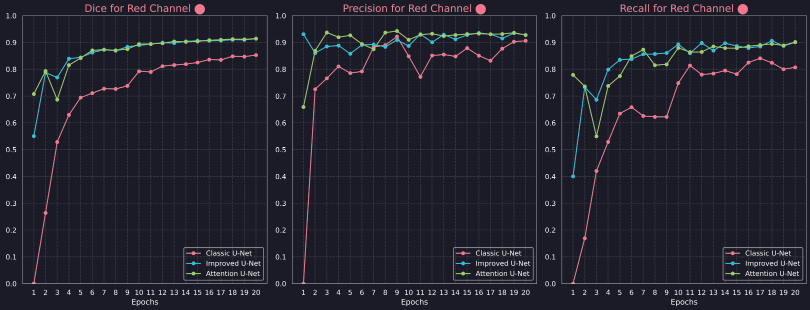
Precision

prediction quality

$$\text{Recall} = \frac{TP}{TP+FN}$$

Recall

prediction quantity



Performance Assessment

$$\text{Dice} = \frac{2\times|X\cap Y|}{|X|+|Y|}$$

Dice Coefficient

"overlap" metric

$$\text{Precision} = \frac{TP}{TP+FP}$$

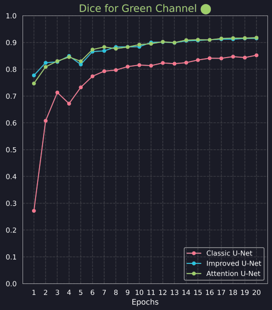
Precision

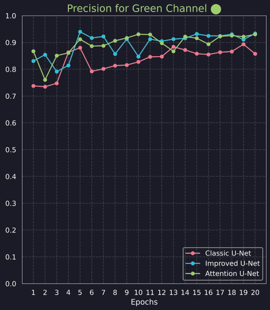
prediction quality

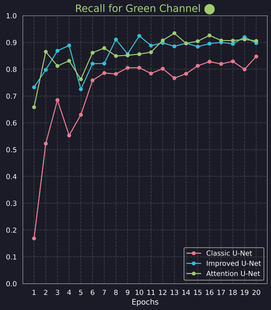
$$\text{Recall} = \frac{TP}{TP+FN}$$

Recall

prediction quantity







Performance Assessment

$$\text{Dice} = \frac{2 \times |X \cap Y|}{|X| + |Y|}$$

Dice Coefficient

"overlap" metric

$$\text{Precision} = \frac{TP}{TP+FP}$$

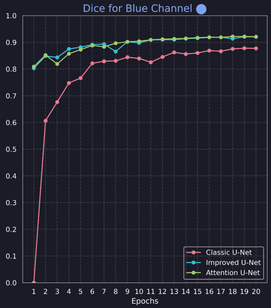
Precision

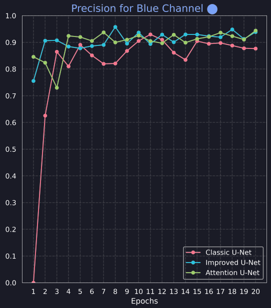
prediction quality

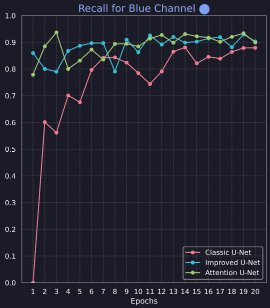
$$\text{Recall} = \frac{TP}{TP+FN}$$

Recall

prediction quantity



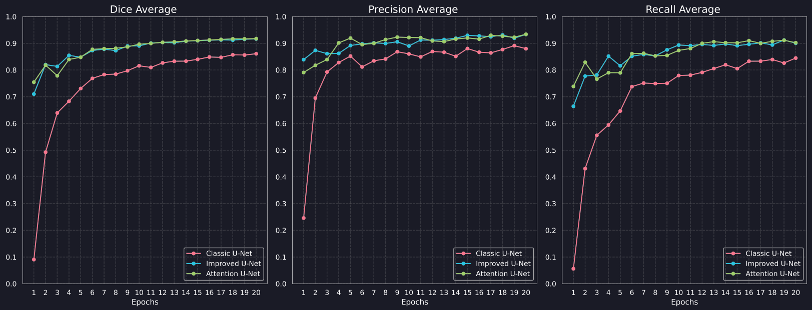




Performance Assessment

$$\text{Dice} = \frac{2 \times |X \cap Y|}{|X| + |Y|} \qquad \text{Precision} = \frac{TP}{TP + FP} \qquad \text{Recall} = \frac{TP}{TP + FN}$$

Dice Coefficient	*Precision*	*Recall*
"overlap" metric	*prediction quality*	*prediction quantity*



Visualizing Attention Maps