

# 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

*A deep learning project for brain tumor classification and segmentation on MRI images using CNN, U-Net, and ViT models.*

## 1. Datasets Description

- Brain Tumor MRI Dataset (*classification*)
- BraTS 2020 Dataset (*segmentation*)

## 2. Classification Task

- Custom CNN
- ViT
- AlexNet
- VGG16

## 3. Segmentation Task

- U-Net Models

## 4. Conclusion

# Datasets

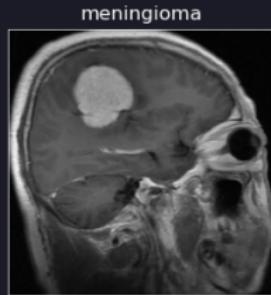
# Brain Tumor MRI Dataset

Classification task dataset

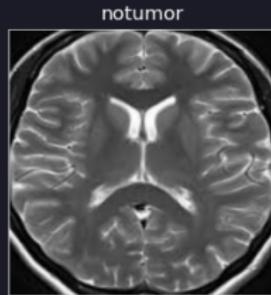
- Combination of three datasets
- 7023 images of human brain MRI images
- Four classes: glioma, meningioma, no-tumor and pituitary



glioma



meningioma



notumor

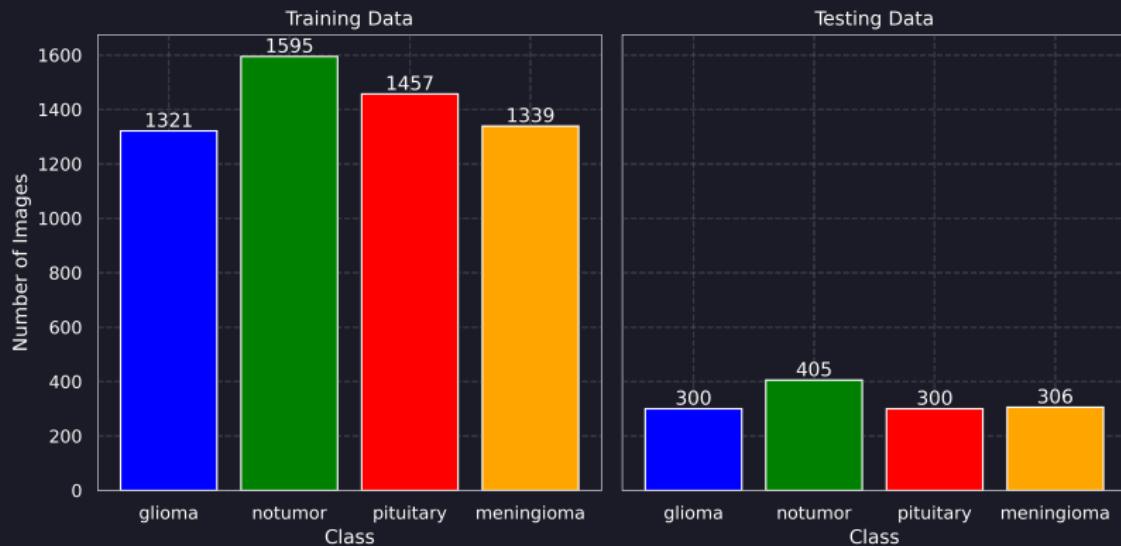


pituitary

## Datasets

# 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  $128 \times 128$  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

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

## BraTS 2020 Dataset

### Segmentation task dataset

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

$$155 \cdot 369 = 57195$$

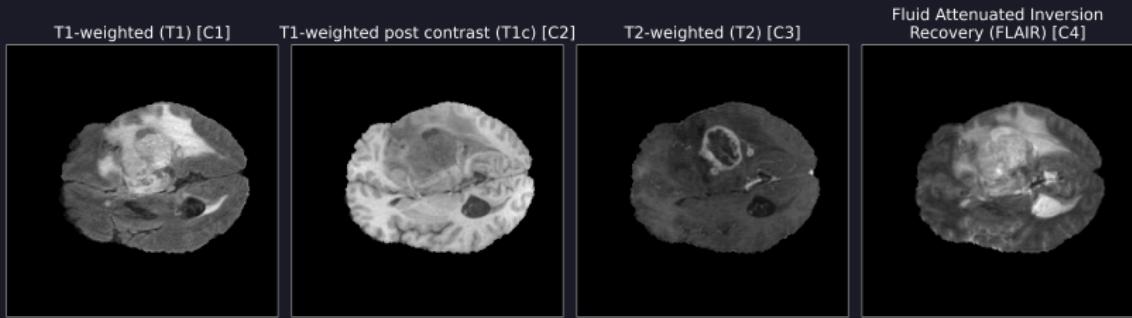
- we used the 50% “*most significant*” slices of the dataset
- we used 90% of data for **training** and 10% for testing

## Datasets

# 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 technique 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*

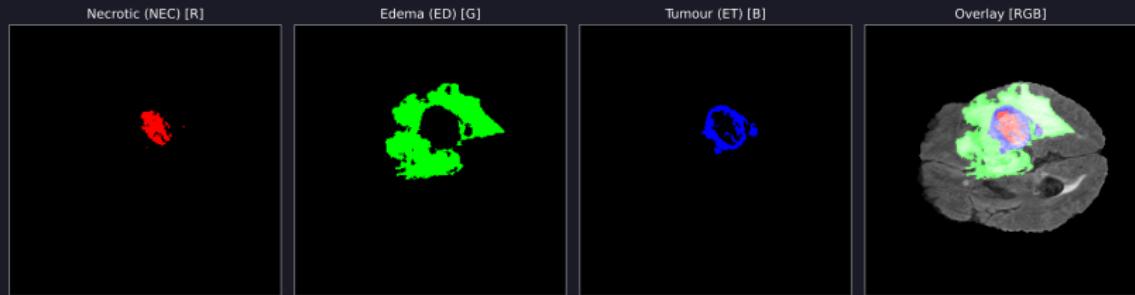


## Datasets

# BraTS 2020 Dataset

Each slice has 3 mask labels (*some might be empty*):

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



# Classification

# Performance Assessment

**Loss Function:** Cross-entropy loss

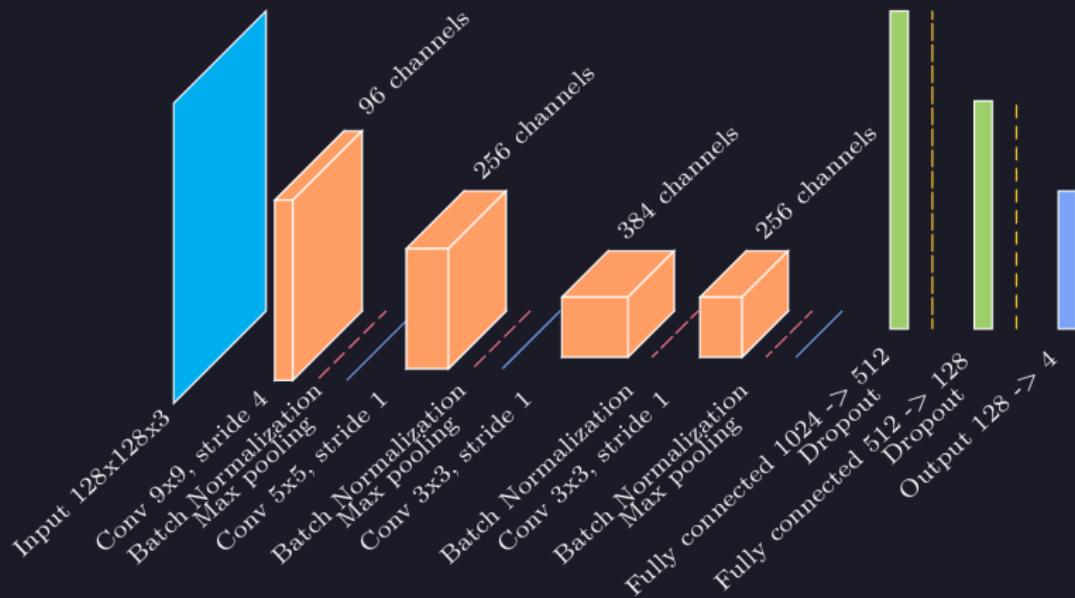
$$L(y, \hat{y}) = - \sum_i y_i \log(\hat{y}_i)$$

**Accuracy:** Number of correct predictions divided by the total number of predictions

**Confidence:** Given by the Softmax function applied to the net output

$$S(x_i) = \frac{e^{x_i}}{\sum_j e^{x_j}}$$

# Custom CNN Architecture



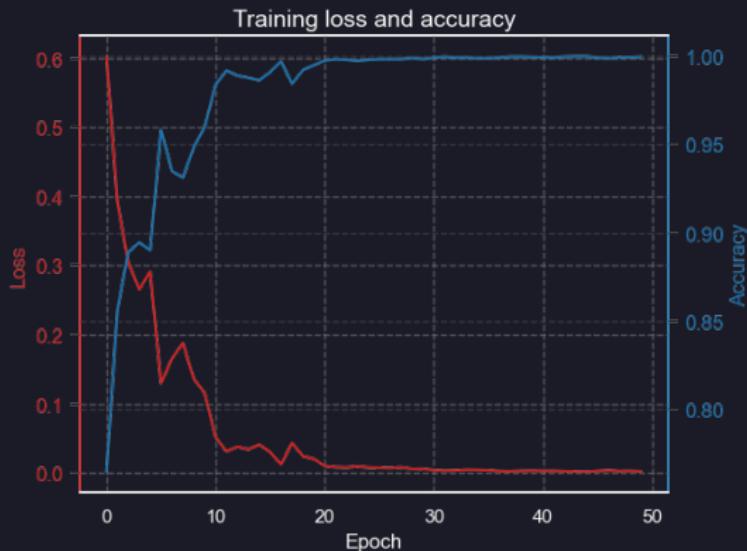
Number of parameters: 3001156

## Training Details

Costum CNN model training parameters:

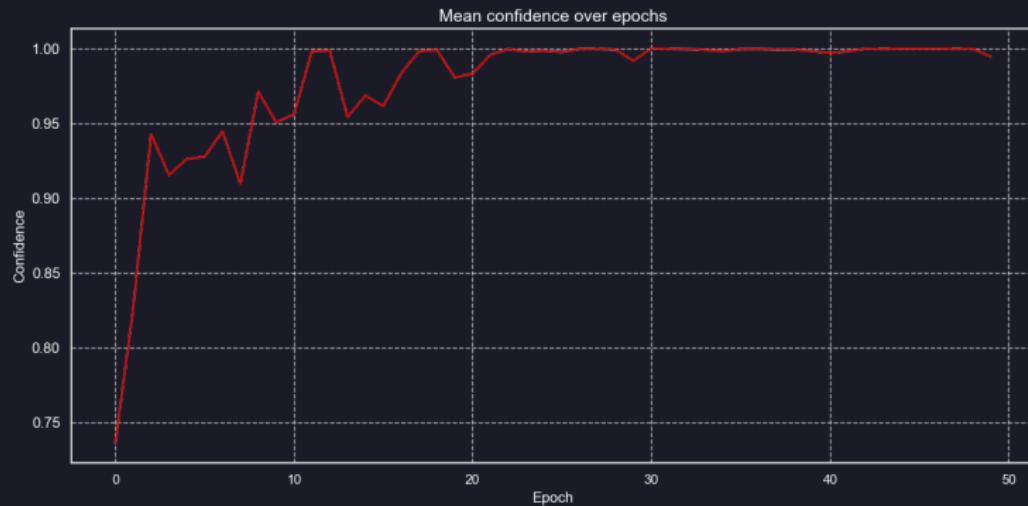
- **Epochs:** 50
- **Optimizer:** Adam (weight decay  $1 \times 10^{-5}$ )
- **Scheduler:** stepLR (step size 10, gamma 0.5)
- **Loss function:** Cross-entropy
- **Learning rate:**  $1 \times 10^{-4}$
- **Batch size:** 64 (both training and validation)
- **Activation function:** Mish
- **Dropout rate:**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

# Training Loss and Accuracy



- Final training loss:  $1.4 \times 10^{-3}$
- Final training accuracy: 99.9%

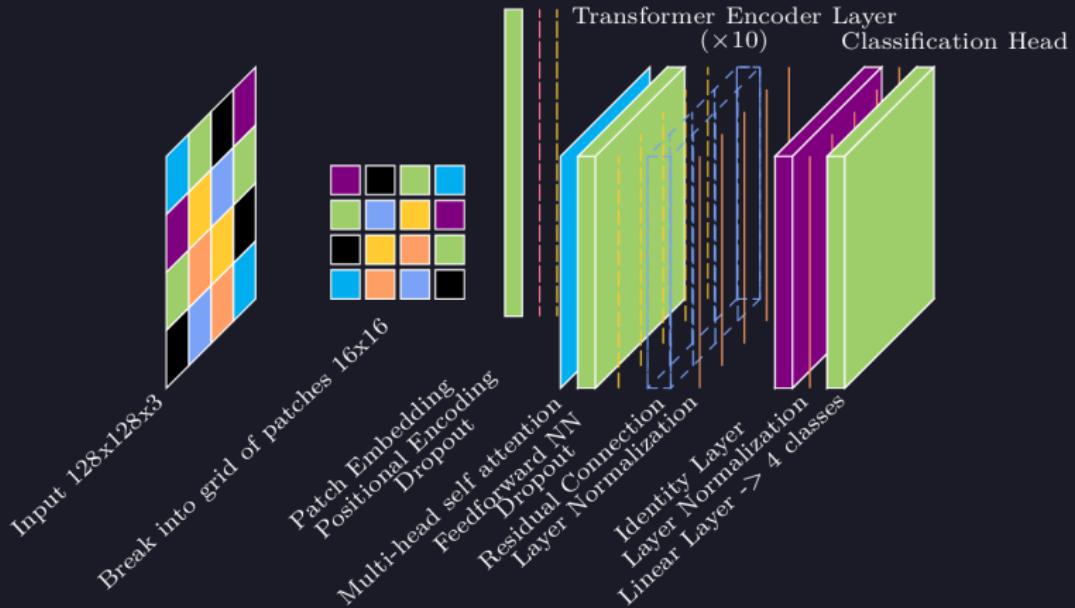
# Confidence and Test Accuracy



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

# Classification

## VIT Architecture



## Training Details

VIT training and model parameters:

- **Epochs:** 50
- **Optimizer:** Adam (weight decay  $1 \times 10^{-5}$ )
- **Scheduler:** stepLR (step size 10, gamma 0.5)
- **Loss function:** Cross-entropy
- **Learning rate:**  $1 \times 10^{-4}$
- **Batch size:** 64 (both training and validation)
- **Activation function:** Mish
- **Dropout rate:** 0.2
- **Image size and Patch size:**  $128 \times 128$ ,  $16 \times 16$
- **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%

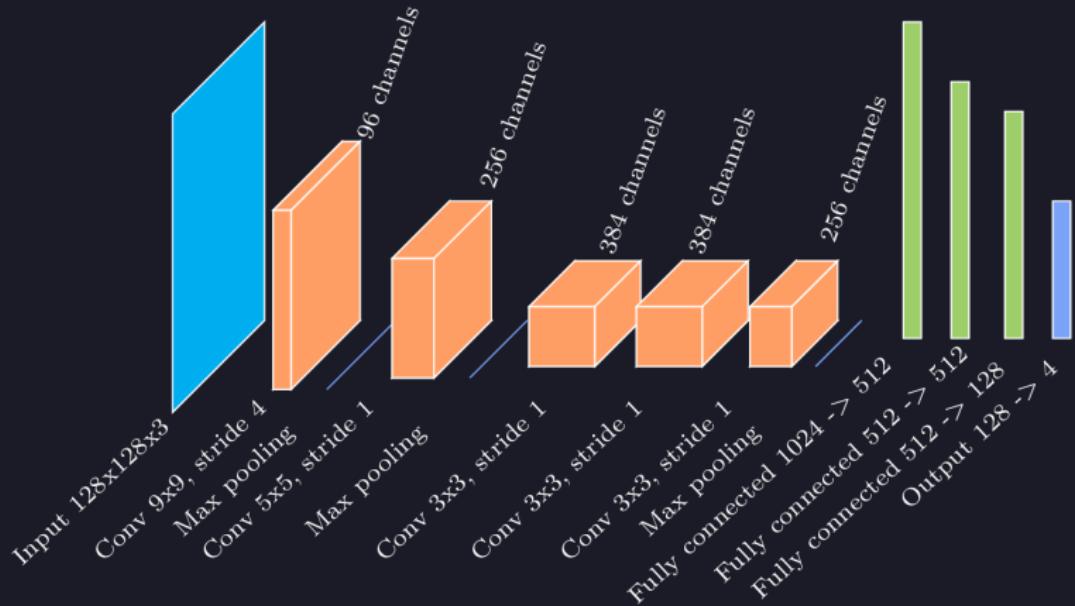
# Confidence and Test Accuracy



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

## Classification

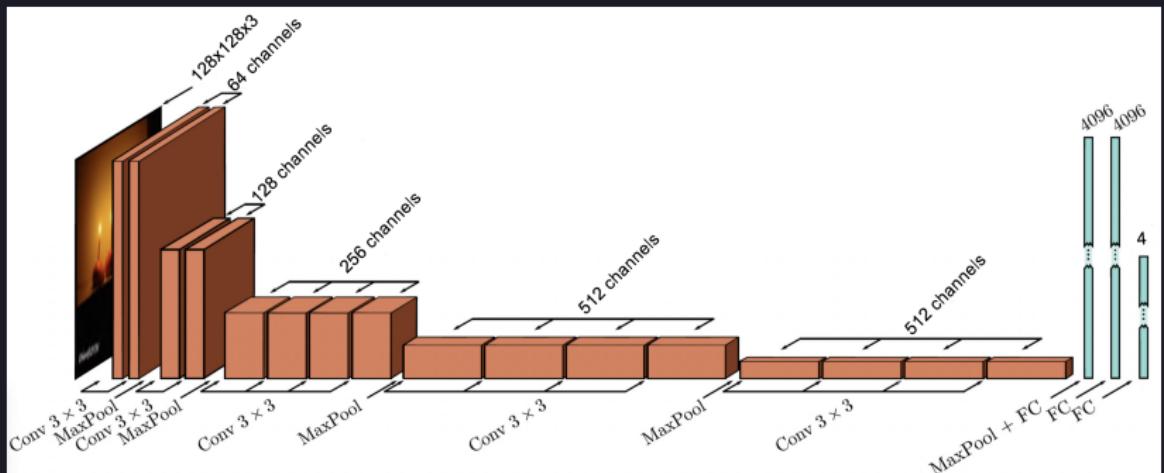
# AlexNet



Number of parameters: 4589316

## Classification

# VGG



Number of parameters: 65070916

Dropout rate: 0.5

# Setup Differences

Model	Data augmentation	Scheduler	Activation	L2 regularization
CustomCNN	Yes ✓	Yes ✓	<i>Mish</i>	Yes ✓
AlexNet	No ✗	Yes ✓	<i>ReLU</i>	Yes ✓
VGG16	No ✗	No ✗	<i>ReLU</i>	No ✗
VIT	Yes ✓	Yes ✓	<i>Mish</i>	Yes ✓

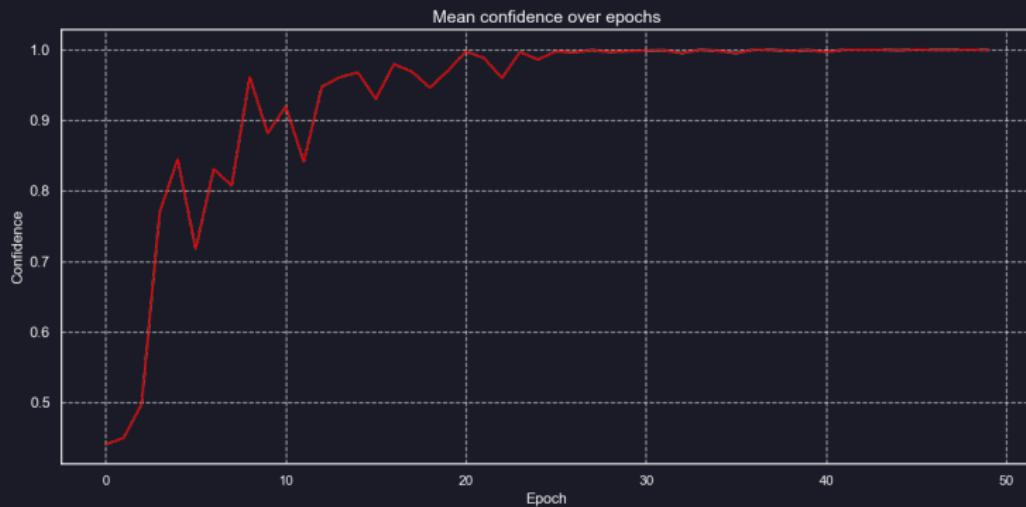
- All the other hyperparameters and settings are the same for all models (*batch size, optimizer, epochs, etc...*)
- Note that the **CustomCNN** is the one with less parameters (3,001,156) while **VGG16** is the one with more parameters(65,070,916)
- **VGG16** also has the highest dropout rate (0.5)

# 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



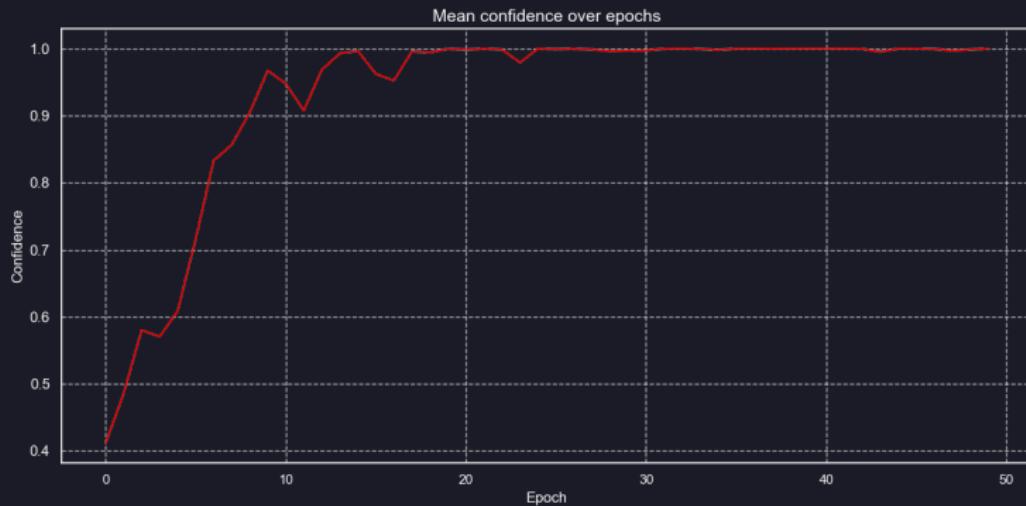
- Final training confidence: 99.9%
- Final test confidence: 96.5%
- Final test accuracy: 90%

# 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	$1.4 \cdot 10^{-3}$	99%	100%
AlexNet	$1.2 \cdot 10^{-3}$	99%	99.9%
VGG16	$8.9 \cdot 10^{-6}$	99%	100%
VIT	0.27	90%	96.1%



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

## Classification

# Focus on Accuracy



# Test Performance Comparison

Model	Accuracy	Confidence
CustomCNN	99%	100%
AlexNet	90%	96.5%
VGG16	95%	98.0%
VIT	88%	93.3%



Note that these are the values reached after the **last epoch**.

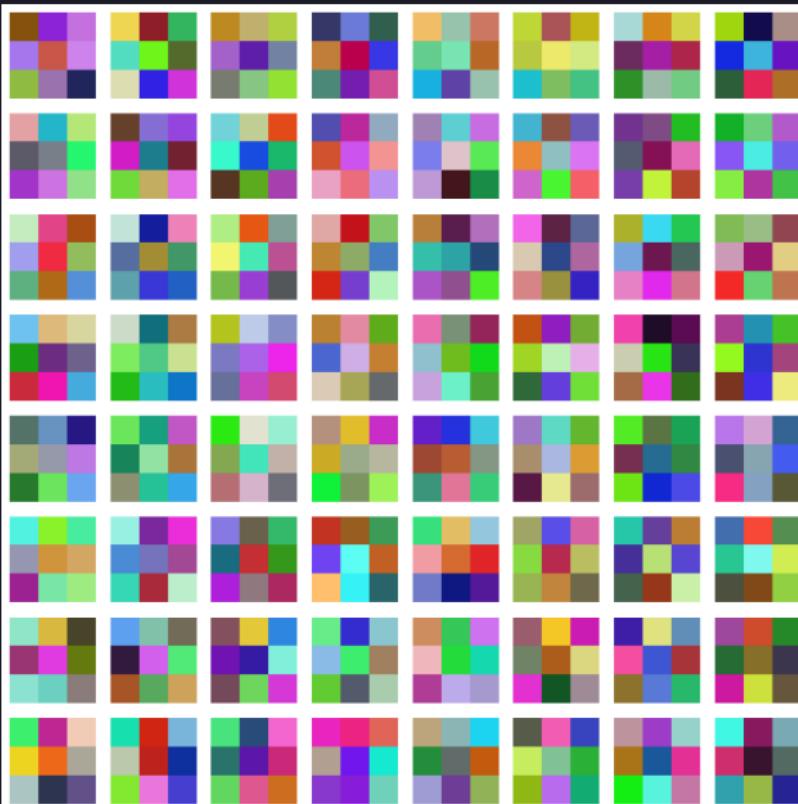
# Visualizing the first layer filters, CustomCNN



# Visualizing the first layer filters, AlexNet



# Visualizing the first layer filters, VGG16



# Segmentation

# 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

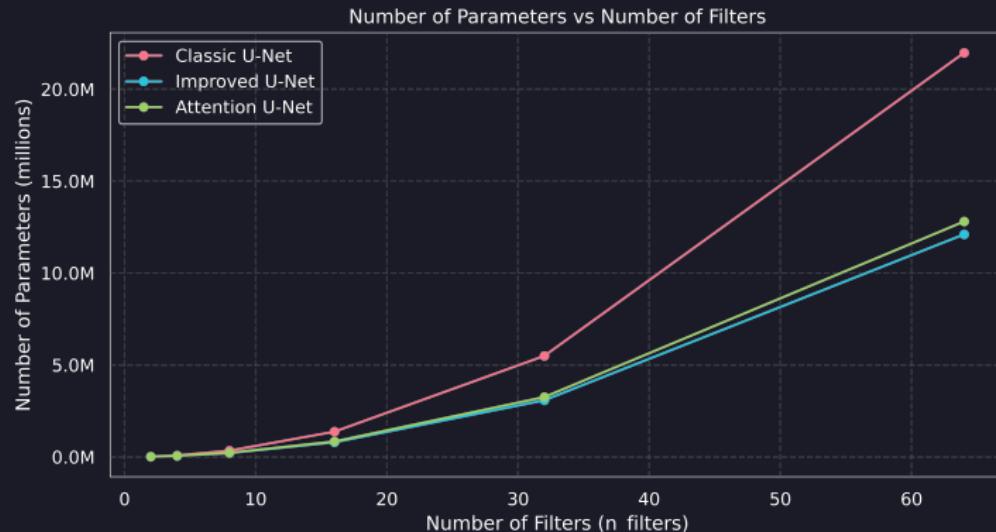
3 models for the segmentation task:

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

# U-Net Models

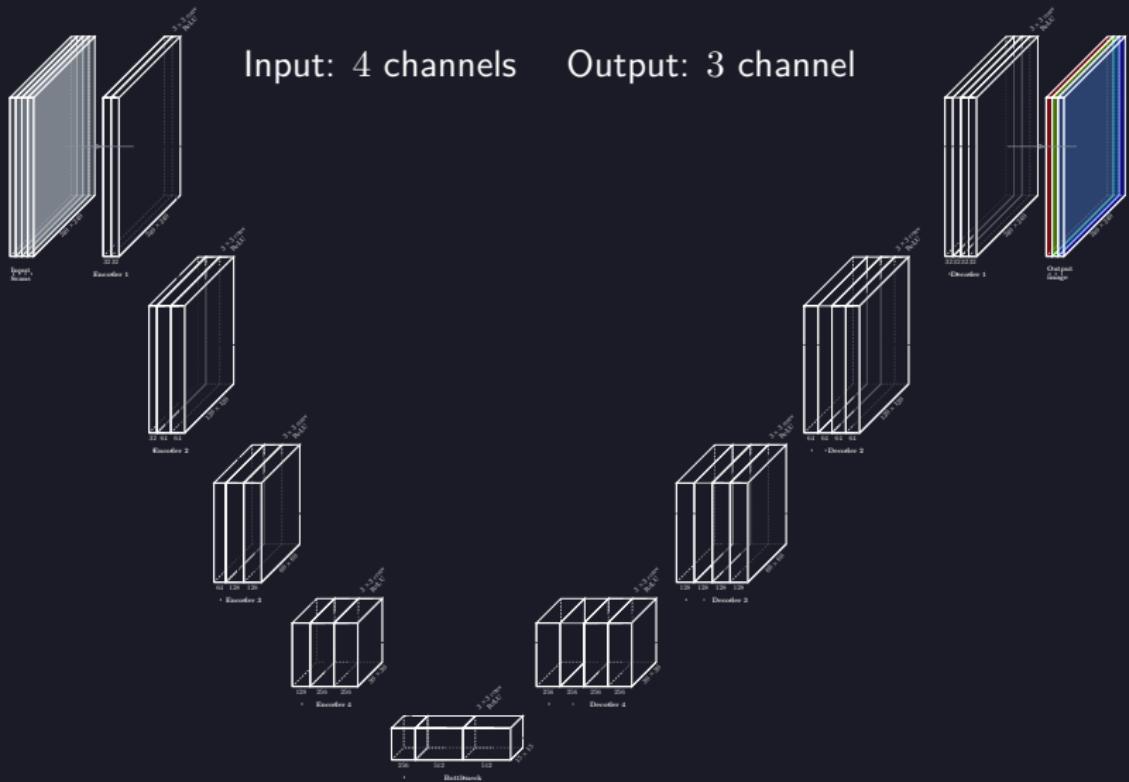
3 models for the segmentation task:

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



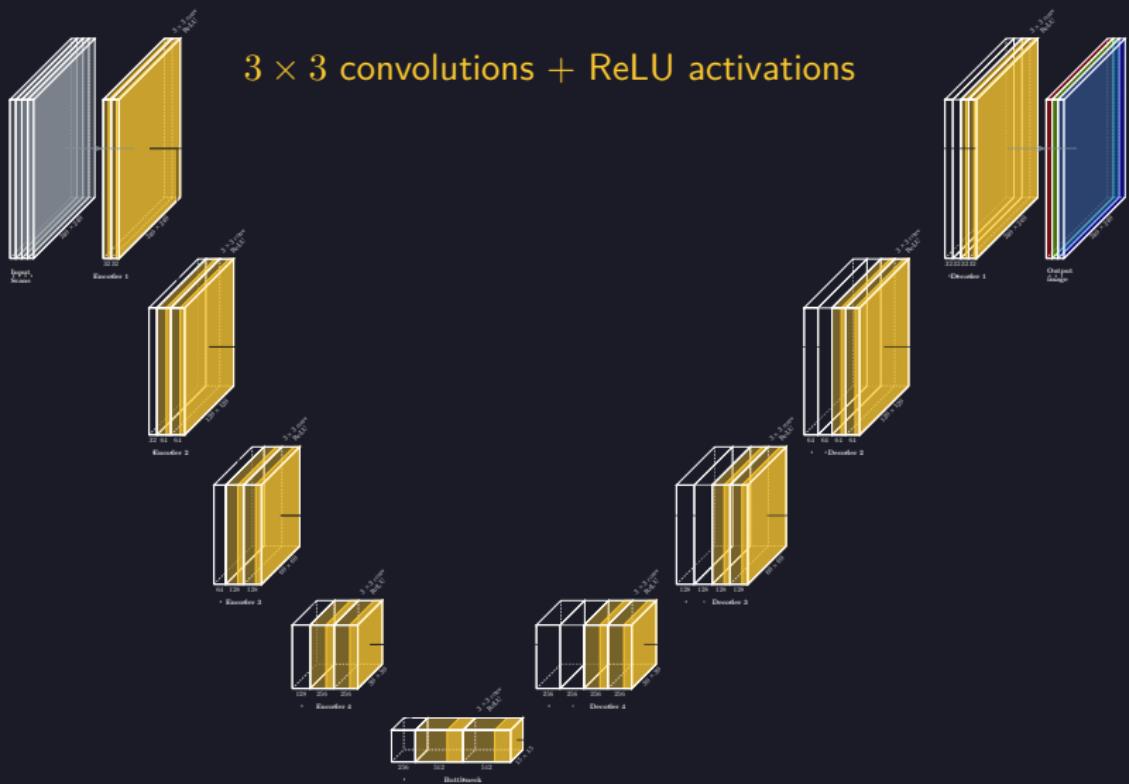
# Segmentation

## Classic U-Net



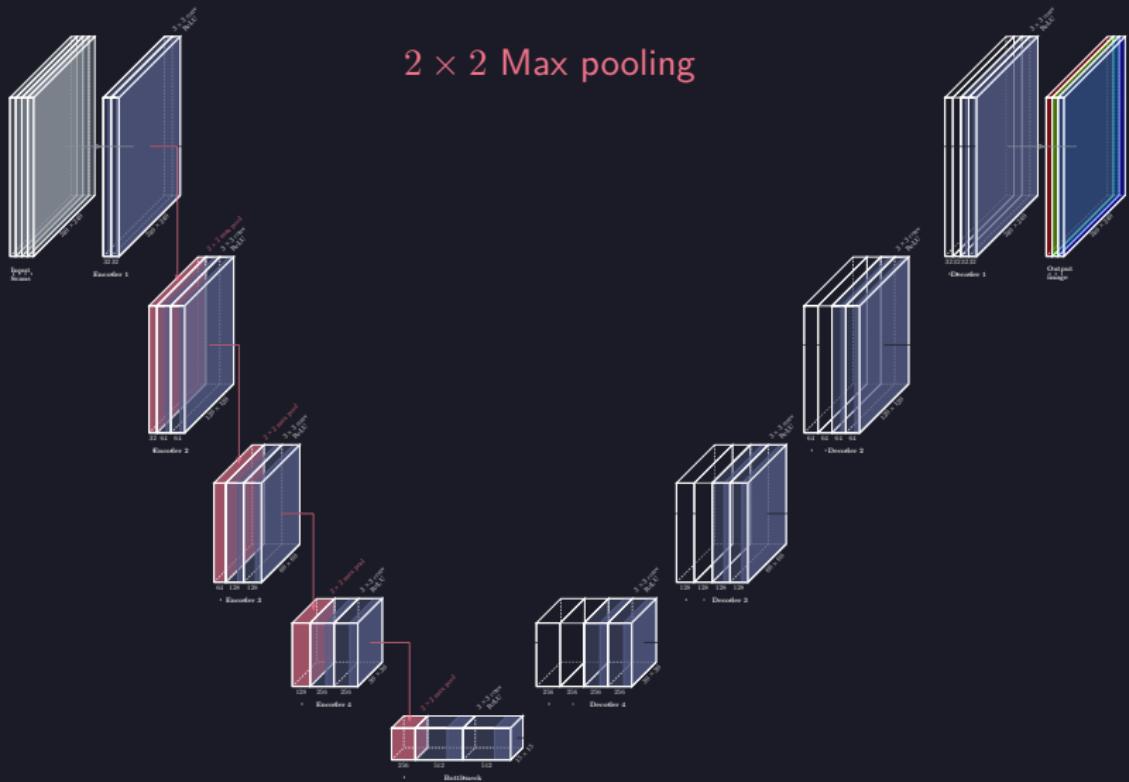
# Segmentation

## Classic U-Net



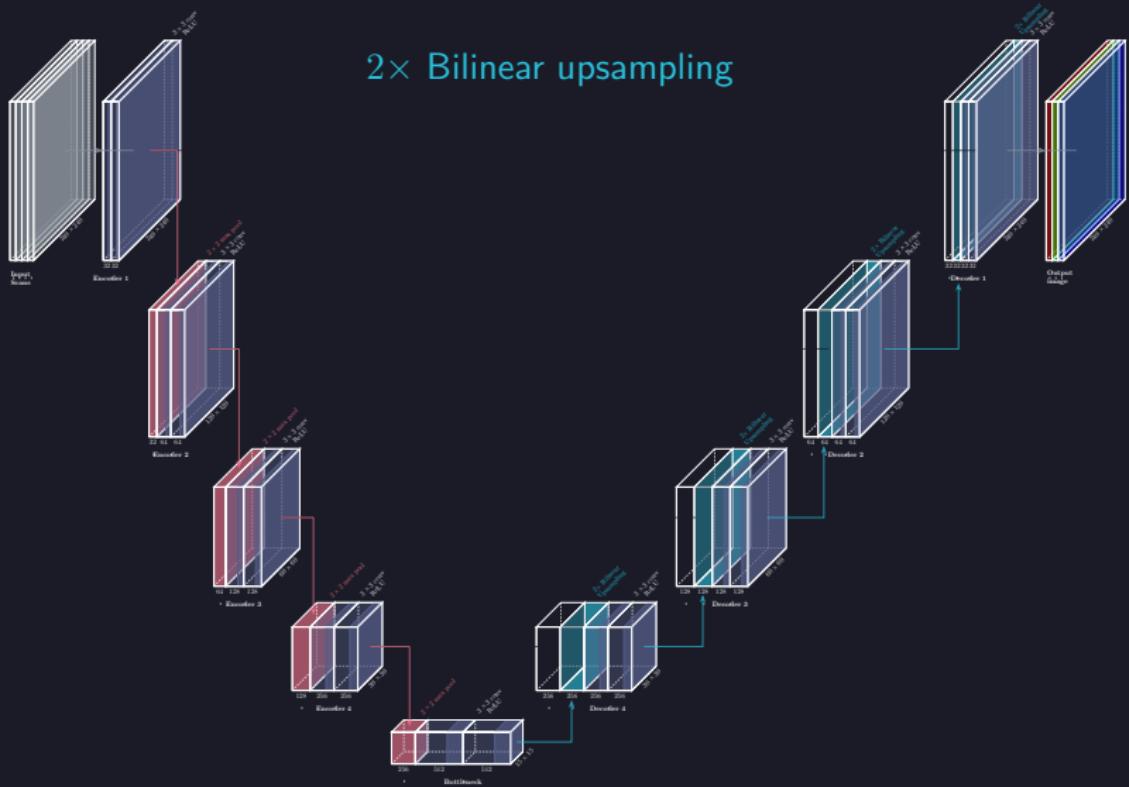
# Segmentation

## Classic U-Net



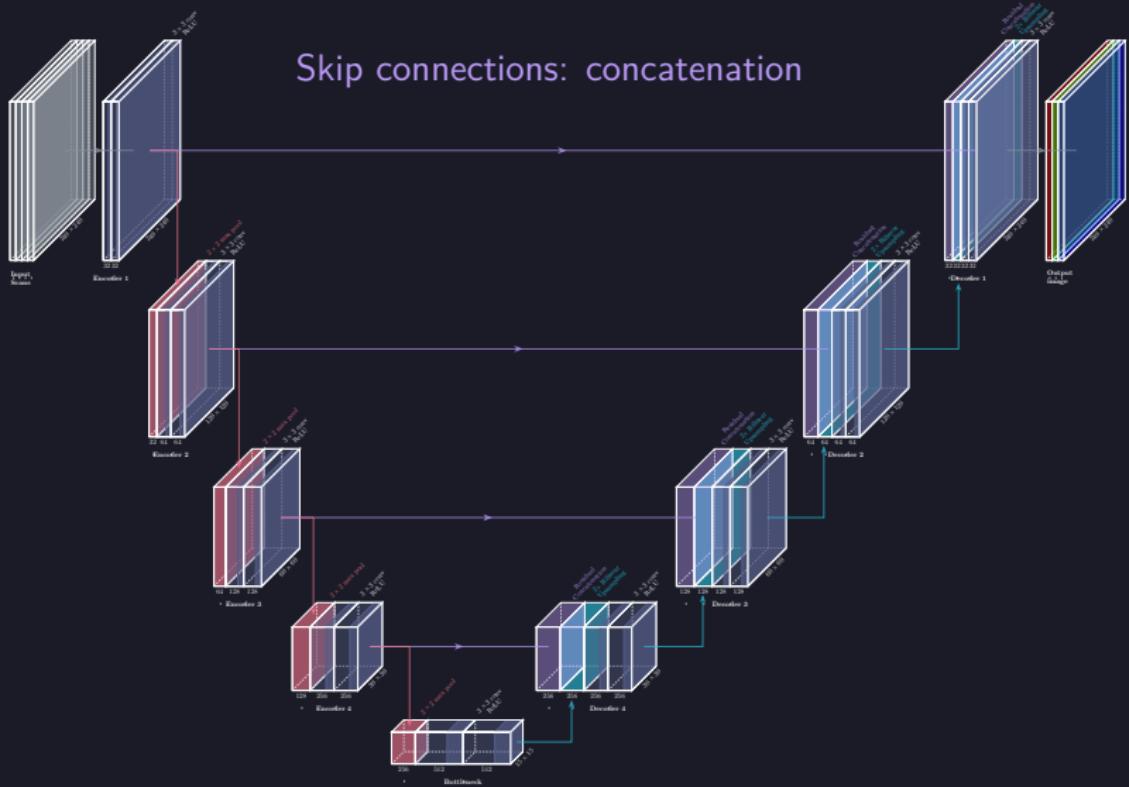
# Segmentation

## Classic U-Net



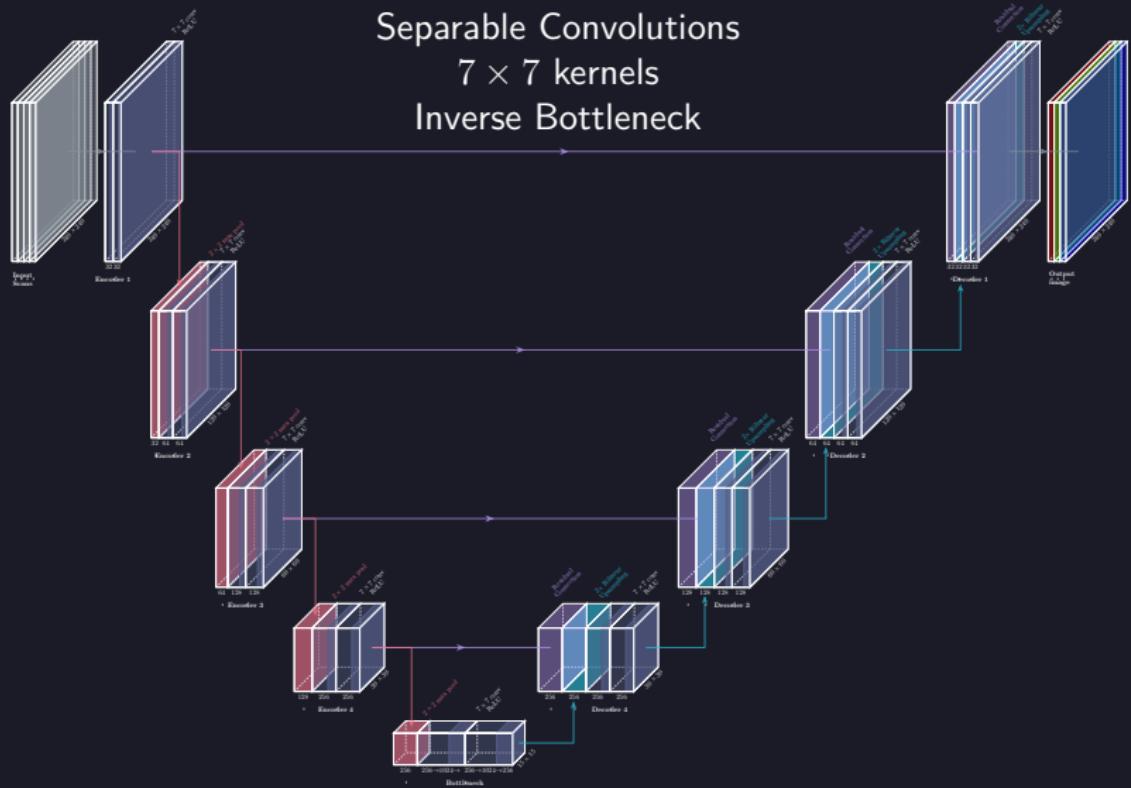
# Segmentation

## Classic U-Net



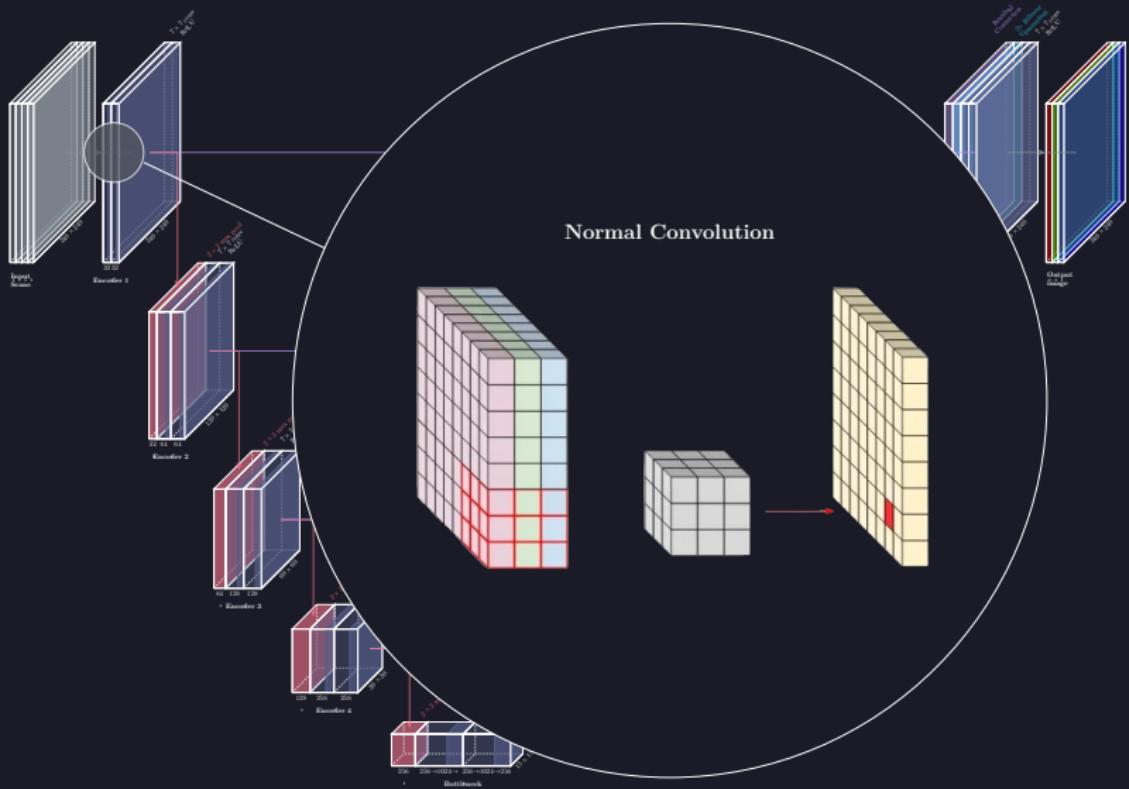
# Segmentation

## Improved U-Net



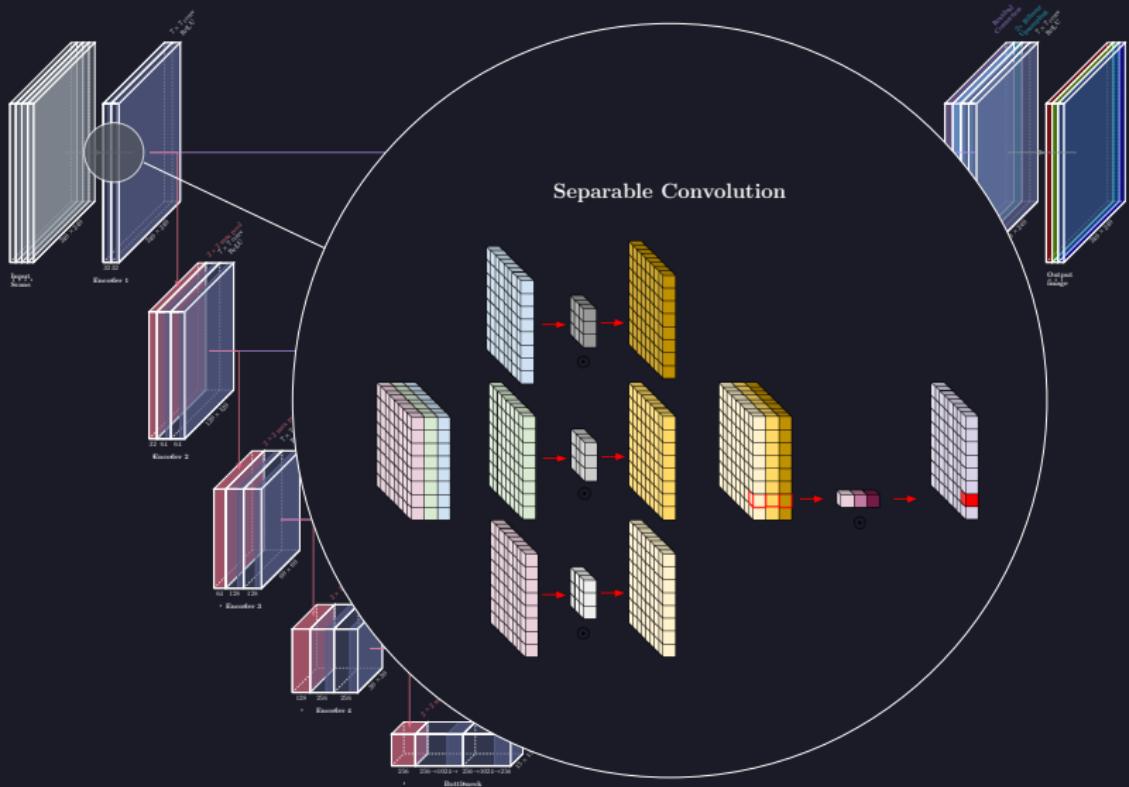
# Segmentation

## Improved U-Net



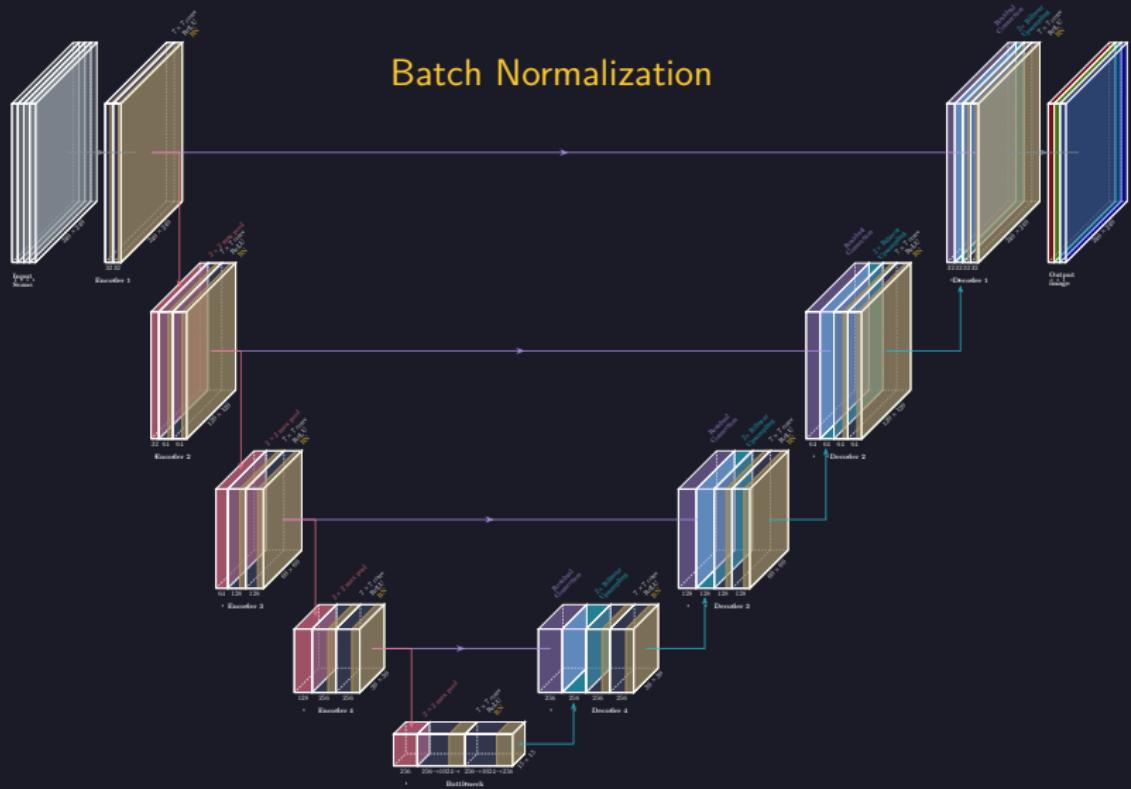
# Segmentation

## Improved U-Net



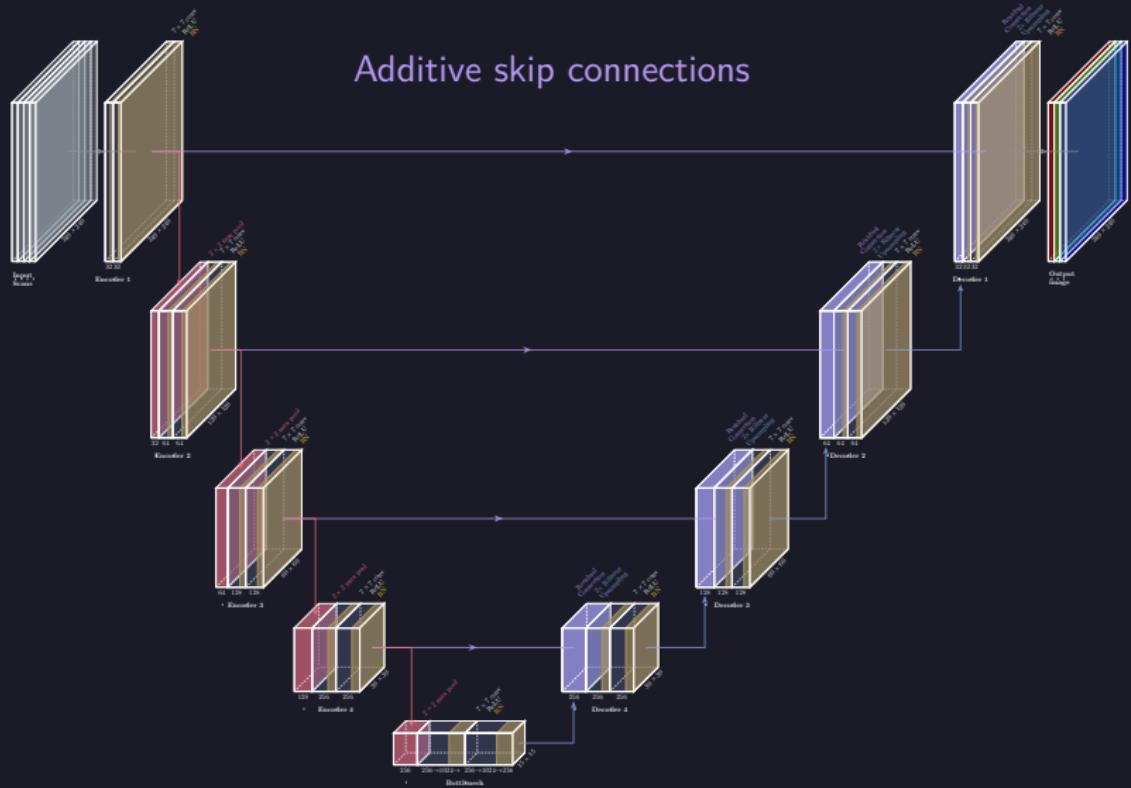
# Segmentation

## Improved U-Net



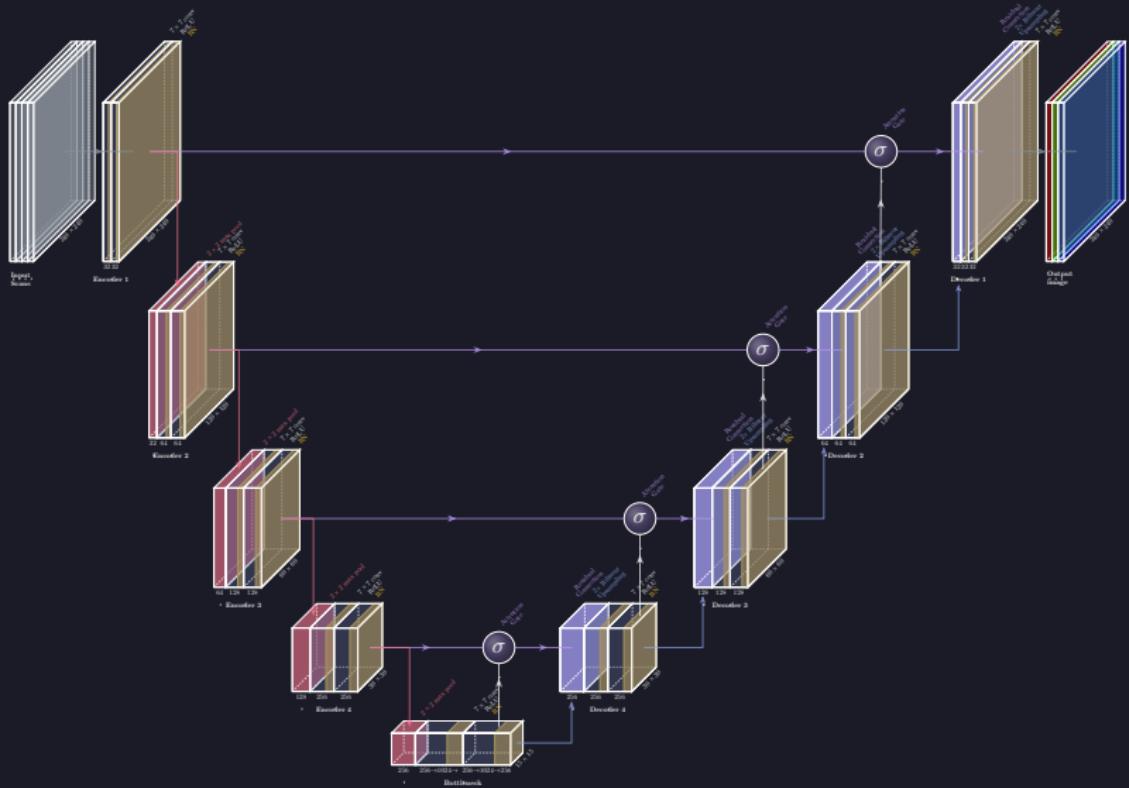
# Segmentation

## Improved U-Net



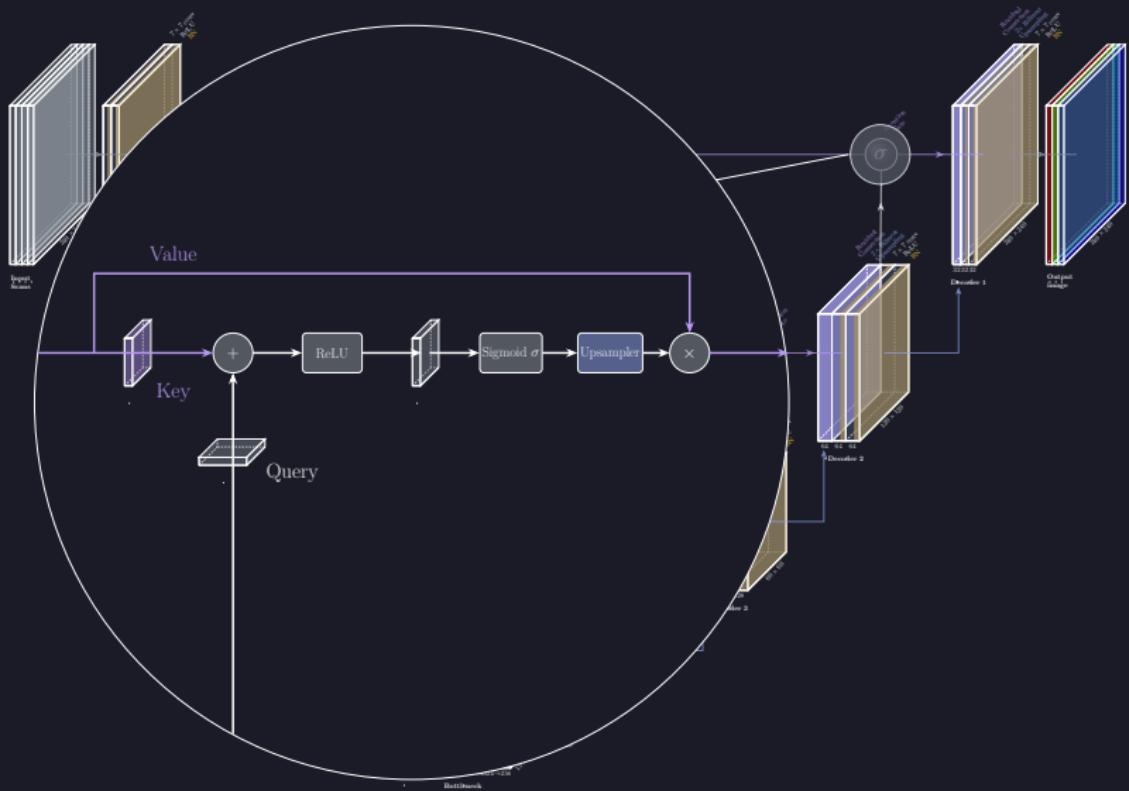
# Segmentation

## Attention U-Net



# Segmentation

## Attention U-Net



# Training Details

U-Net models training parameters:

- **Epochs:** 20
- **Optimizer:** Adam (with weight decay  $1 \times 10^{-2}$ )
- **Scheduler:** Exponential Decay ( $\gamma = 0.9$ )
- **Loss function:** BCE with Logits Loss:

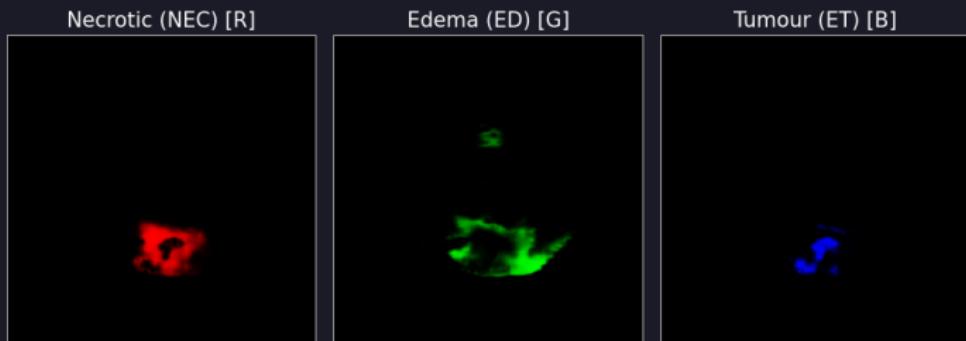
$$\ell(y, \hat{y}) = -[y \log(\sigma(\hat{y})) + (1 - y) \log(1 - \sigma(\hat{y}))]$$

- **Learning rate:**  $2 \times 10^{-3}$
- **Batch size:** 32 (both training and validation)
- **First encoder filters:** 32
- **Image size:**  $240 \times 240$

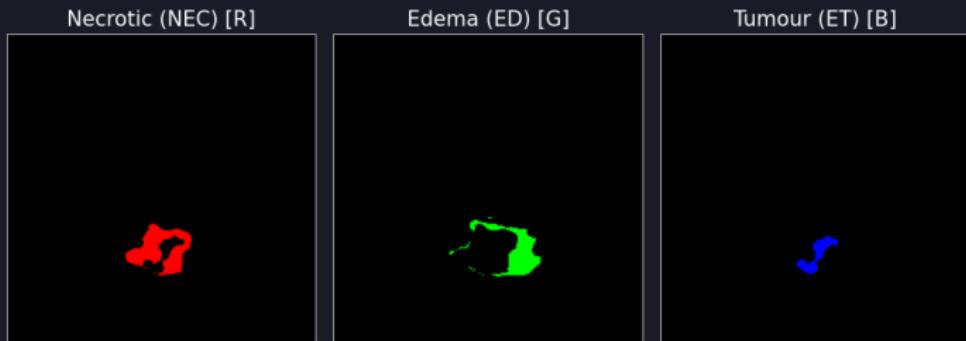
## Segmentation

# Visualizing a prediction

Predicted Mask Channels [RGB]



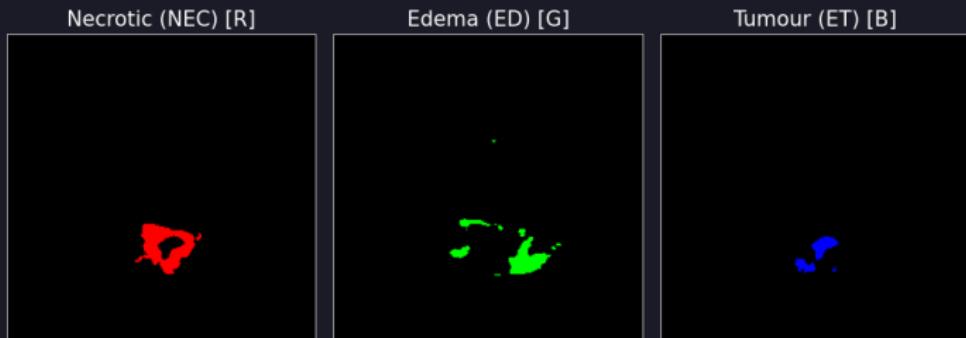
Ground Truth Mask Channels [RGB]



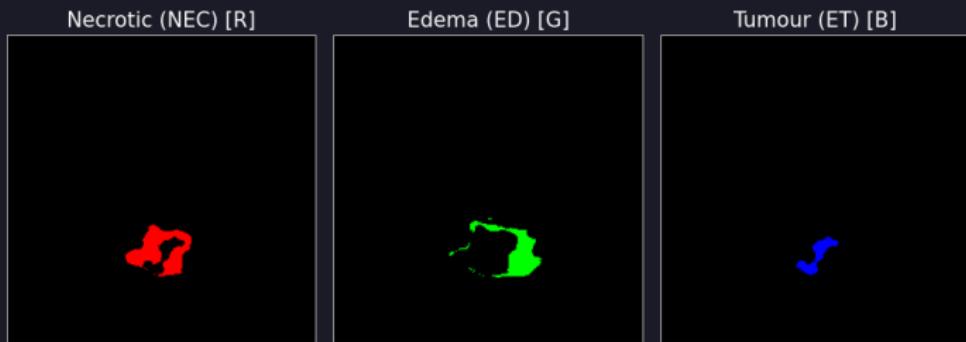
## Segmentation

# Visualizing a prediction

Binarized Predicted Mask Channels [RGB]



Ground Truth Mask Channels [RGB]



# Segmentation

## Performance Assessment

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

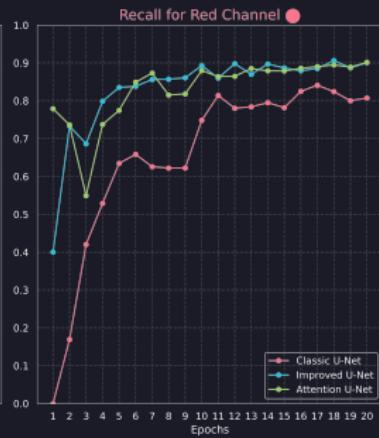
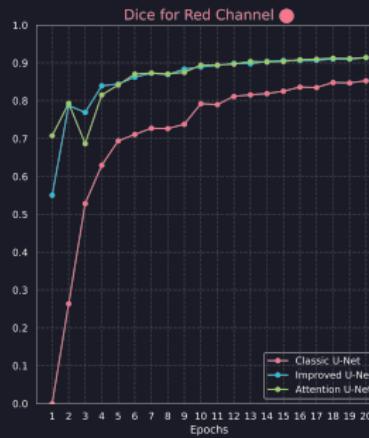
Dice Coefficient  
"overlap" metric

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

Precision  
prediction quality

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

Recall  
prediction quantity



# Segmentation

## Performance Assessment

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

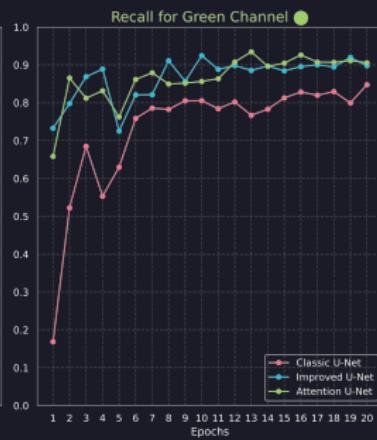
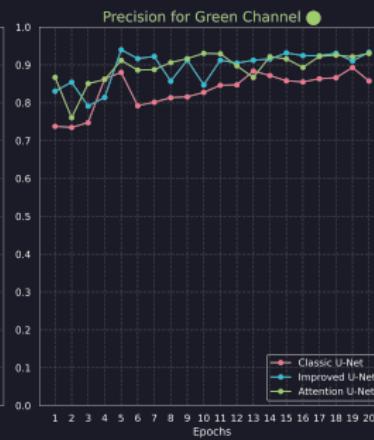
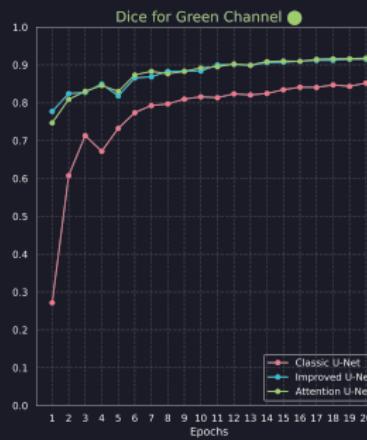
Dice Coefficient  
"overlap" metric

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

Precision  
prediction quality

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

Recall  
prediction quantity



# Segmentation

## Performance Assessment

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

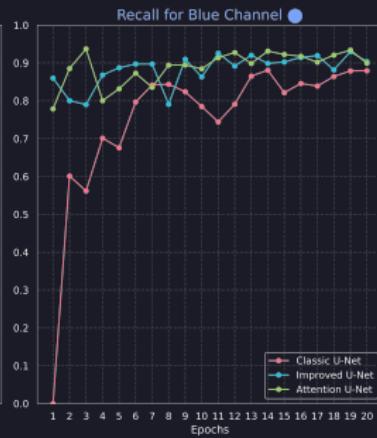
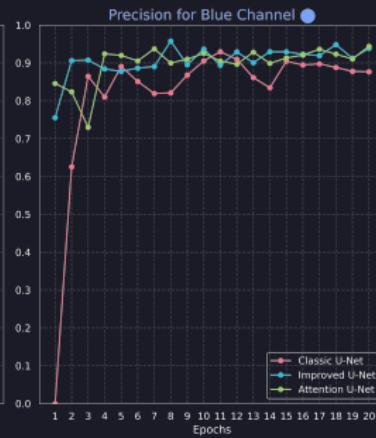
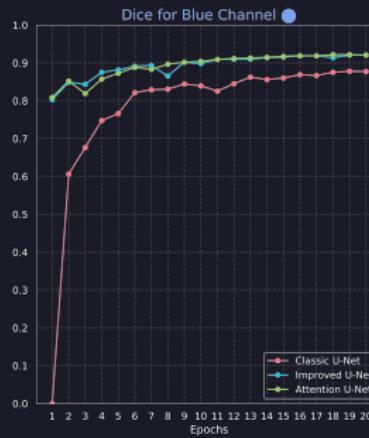
Dice Coefficient  
“overlap” metric

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

Precision  
prediction quality

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

Recall  
prediction quantity



# Segmentation

## Performance Assessment

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

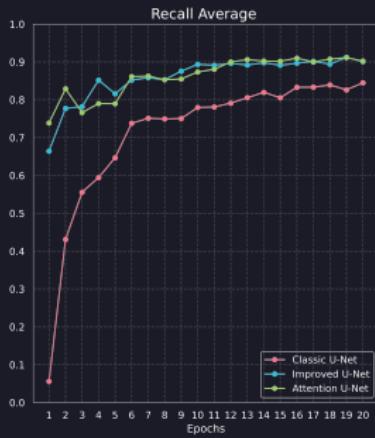
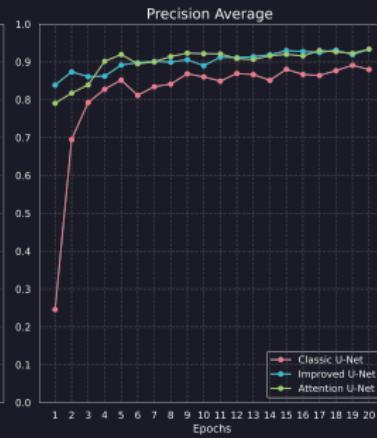
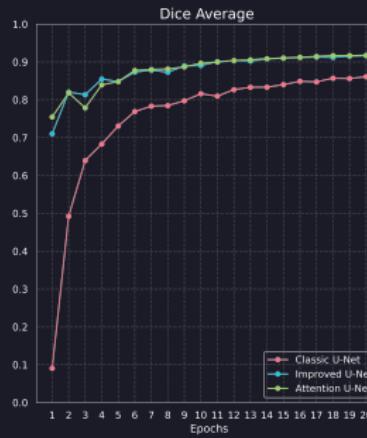
Dice Coefficient  
“overlap” metric

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

Precision  
prediction quality

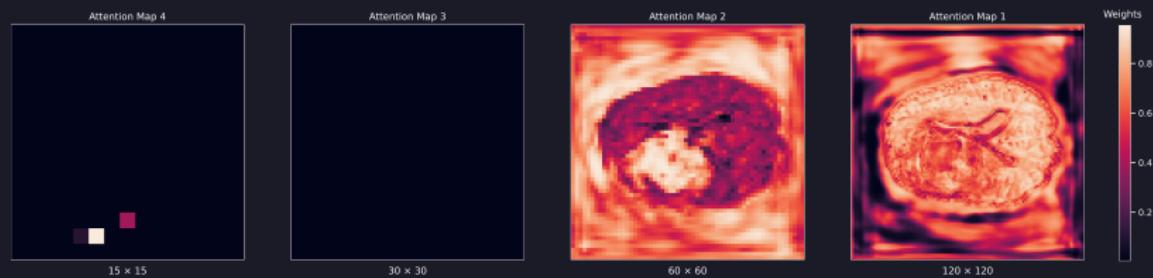
$$\text{Recall} = \frac{TP}{TP + FN}$$

Recall  
prediction quantity



## Segmentation

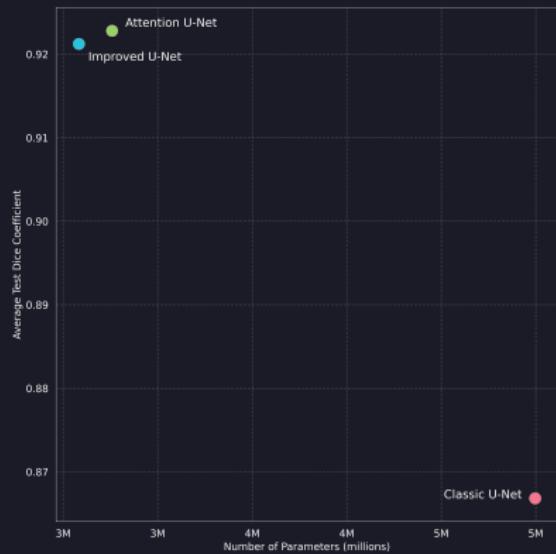
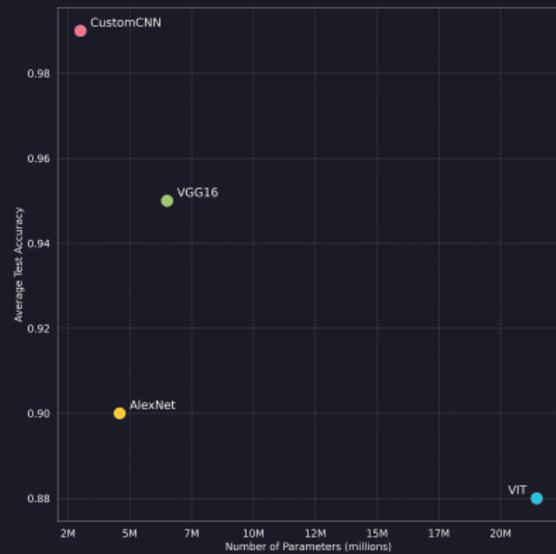
# Visualizing Attention Maps



# Conclusion

## Conclusion

# Conclusion



# Conclusion

Possible improvements:

- Use original image size for better classification results
- Use larger dataset to train the VIT model
- Enlarge the VIT model for better performance
- Attempt transfer learning with VIT model
- Test different architectures for the segmentation task
- Fully exploit segmentation dataset
- Perform complete hyperparameter search for the segmentation models
- Use metadata information to predict patient's survival days

## References

# References

- [1] B. H. Menze et al. "The Multimodal Brain Tumor Image Segmentation Benchmark (BRATS)", IEEE Transactions on Medical Imaging 34(10), 1993-2024 (2015) DOI: 10.1109/TMI.2014.2377694
- [2] S. Bakas et al. "Advancing The Cancer Genome Atlas glioma MRI collections with expert segmentation labels and radiomic features", Nature Scientific Data, 4:170117 (2017) DOI: 10.1038/sdata.2017.117
- [3] S. Bakas et al. "Identifying the Best Machine Learning Algorithms for Brain Tumor Segmentation, Progression Assessment, and Overall Survival Prediction in the BRATS Challenge", arXiv preprint arXiv:1811.02629 (2018)
- [4] S. Bakas et al. "Segmentation Labels and Radiomic Features for the Pre-operative Scans of the TCGA-GBM collection", The Cancer Imaging Archive, 2017 DOI: 10.7937/K9/TCIA.2017.KLXWJJ1Q
- [5] S. Bakas et al. "Segmentation Labels and Radiomic Features for the Pre-operative Scans of the TCGA-LGG collection", The Cancer Imaging Archive, 2017 DOI: 10.7937/K9/TCIA.2017.GJQ7R0EF
- [6] Alex Krizhevsky et al. "ImageNet classification with deep convolutional neural networks", Commun. ACM 60, 6 (June 2017), 84–90,  
<https://doi.org/10.1145/3065386>

## References

# References

- [7] A. Dosovitskiy et al. "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale", International Conference on Learning Representations  
<http://openreview.net/forum?id=YicbFdNTTy>
- [8] Omer, A.A.M. "Image Classification Based on Vision Transformer", Journal of Computer and Communications, 12, 49-59 (2024)  
<https://doi.org/10.4236/jcc.2024.124005>
- [9] Ronneberger, O., Fischer, P., & Brox, T. "U-Net: Convolutional Networks for Biomedical Image Segmentation", In Nassir Navab, Joachim Hornegger, William M. Wells, & Alejandro F. Frangi (Eds.), Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015, 234–241  
[https://doi.org/10.1007/978-3-319-24574-4\\_28](https://doi.org/10.1007/978-3-319-24574-4_28)
- [10] Chollet, F. "Xception: Deep Learning with Depthwise Separable Convolutions", CoRR, abs/1610.02357 <http://arxiv.org/abs/1610.02357>
- [11] Liu, Z., Mao, H., Wu, C.-Y., Feichtenhofer, C., Darrell, T., & Xie, S. "A ConvNet for the 2020s", arXiv preprint arXiv:2201.03545 <https://arxiv.org/abs/2201.03545>
- [12] Sandler, M. et al. "Inverted Residuals and Linear Bottlenecks: Mobile Networks for Classification, Detection and Segmentation", CoRR, abs/1801.04381  
<http://arxiv.org/abs/1801.04381>
- [13] Oktay, O. et al. "Attention U-Net: Learning Where to Look for the Pancreas", CoRR, abs/1804.03999 <http://arxiv.org/abs/1804.03999>