Project Progress Report

Image Segmentation Using Transformer like U-Net with Attention

Moaaz Ur Rehman Azhar Khokhar - 0081002

Project Introduction

I added a detailed introduction in the project proposal and here I will summarize it. I selected an image segmentation problem using U-Net by embedding Attention based mechanism to solve the problem. I mentioned in the proposal that I will be implementing U-Transformer (Petit et al., 2021) - a U-Net based implementation by embedding self and cross attention modules at decoder layers - to segment brain tumors from MRI images. Moreover, I am going to compare U-Transformer with Attention U-Net (Ozan et al., 2018) - embeds attention gates at each decoder layer -, which was not in my initial plan. Initially the attention for images was implemented by (Dosovitskiy et al., 2020), and commonly known as Vision Transformer (ViT). U-Transformer is inspired by the original Transformer (Vaswani et al., 2017).

Progress

I have implemented three types of U-Net (from scratch) in the project:

- U-Net (Basic U-Net)
- Attention U-Net (Ozan et al., 2018)
- U-Transformer (Petit et al., 2021)

Architectures of these models are in the next section. I prepared the dataset by a python script which creates annotation files. I have written code for all three U-Nets and ran on the dataset mentioned in the dataset section. I wrote all the code from scratch but for positional encoding I borrowed `positional_encoding()` function from the Transformer tutorial on Tensorflow website (https://www.tensorflow.org/text/tutorials/transformer).

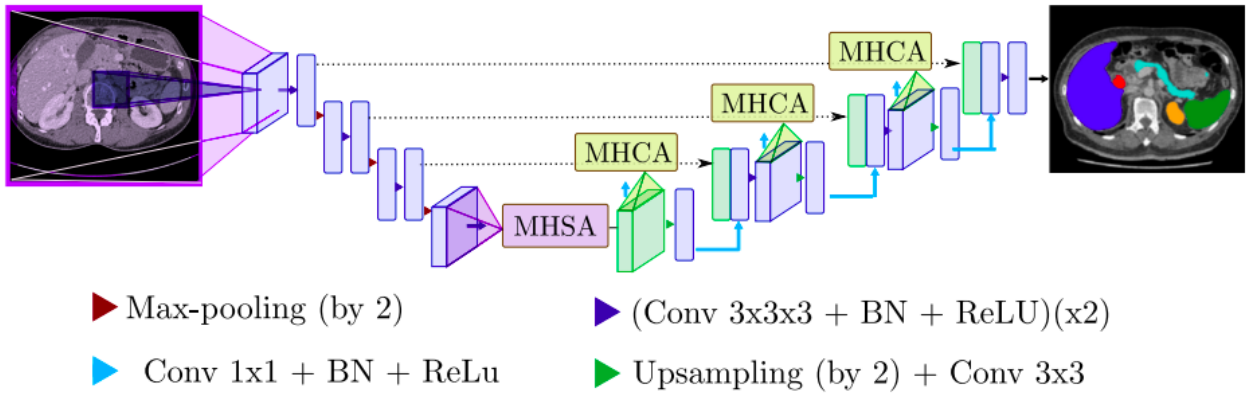
U-Transformer had only one unofficial implementation which was not correct. The developer did not implement multihead attention and moreover he/she implemented the wrong activation function as the authors mentioned of implementing Softmax in Cross Attention module.

Attention U-Net has an official (and several unofficial implementations), which was very complex and generic for their purpose. I implemented this model by myself from scratch and simplified and specialized for my use case.

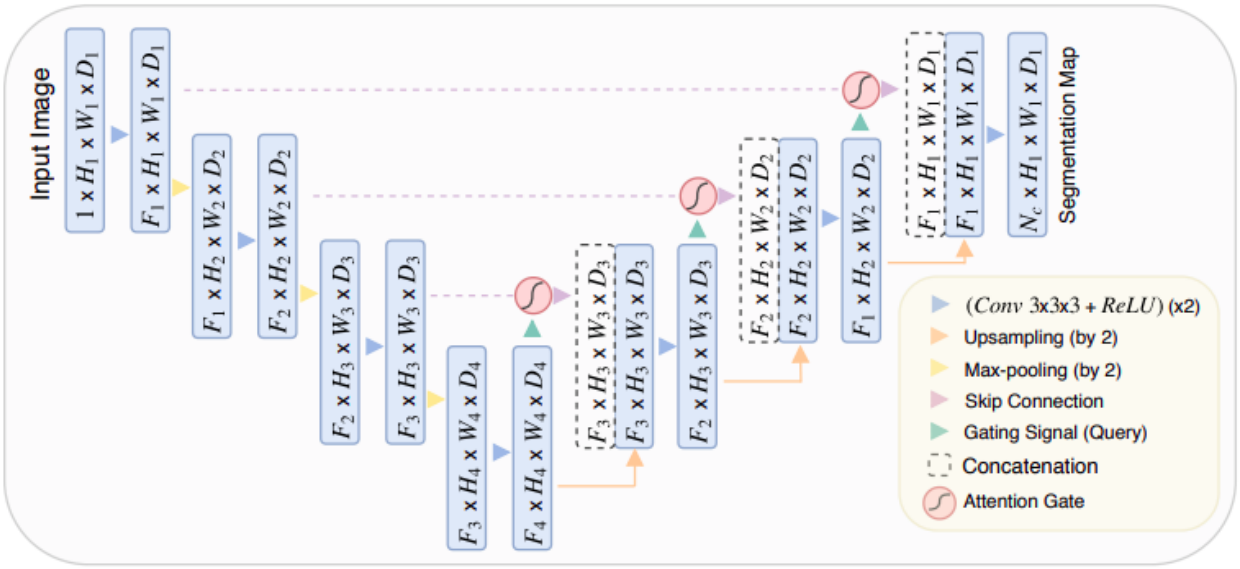
I ran all models on my personal computer GPU for 5 epochs to validate our models. I faced some difficulties which I shall mention later. I have used BCE with Dice Loss as loss function as I cannot simply depend on BCE loss for segmentation tasks. Just after 5 epochs all three models give a Dice score in the range of [0.53, 0.55] with accuracy above 0.95. U-Transfomer has ~35M parameters while Attention U-Net has ~33M parameters. I checked it by "`torchsummary`" package.

Architecture and Methodology

The architecture which I have implemented for the project for **U-Transformer** is given in the following figure.



U-Transformer (Petit et al. 2021) augments U-Nets with transformers to model long-range contextual interactions. The Multi-Head Self-Attention (MHSA) module at the end of the U-Net encoder gives access to a receptive field containing the whole image (shown in purple), in contrast to the limited U-Net receptive field (shown in blue). Multi-Head Cross-Attention (MHCA) modules are dedicated to combine the semantic richness in high level feature maps with the high resolution ones coming from the skip connections.



Above figure is of **Attention U-Net**; at each skip-connection they have embedded Attention gates which highlight bigger values and decrease low values.

I extended both of the architectures by replacing Conv-blocks with residual Conv-blocks in order to increase in test accuracy (IoU). I built all three networks with [32, 64, …, 1024] channels

doubling at each downstep in encoder. I tried to exclude 1024 channels but it decreased the performance of all nets. Moreover, I used **Adam** optimizer with 1e-3 **learning rate** for the above mentioned results. I used the loss function as: `BCE + 1 - DiceLoss`.

Difficulties

While writing MHSA and MHCA, I could not make an intuition for `embed_dim` for the `nn.MultiheadAttention` module of PyTorch. I had discussions with hoca and TAs who helped me in understanding this concept. Moreover, while initializing attention modules, memory overflowed which I solved by decreasing the spatial dimension of input images from 256x256 to 128x128. Furthermore, initially I was using 16 batch size for U-Net but for Attention U-Net and U-Transfomer, I had to decrease batch size to 4 and 2, respectively for it to be fitted in the memory (GPU).

PyTorch's `nn.MultiheadAttention` module has another problem: if you give it `bias=False`, it does not work as it tries to initialize weights for bias. It is not handled and they have an issue on GitHub. It took a lot of time for me to identify and mitigate this error as the error was not descriptive "`Nonetype has no attribute: size()`". Moreover, this module's `forward()` function takes `need_weights` argument, if I set it to `False` it gave error while running for "`torchsummary`" package, which I used to see my model's summary and number of parameters.

Results

As mentioned before, for 5 epochs, from all models I got a Dice score in the range of [0.53, 0.55] with accuracy above 0.95. I got these results without any tuning of hyperparameters. As I wanted to validate these models if they could run without any issue, before I run them to train on the dataset and fine tune them.

Remaining or Additional Work

Following things are in my pipeline:

- Write a solver class as we got in our assignments so that I can easily search for best hyperparameters and best model
- Perform ablation studies to compare performance of all three models on different metrics such as: Accuracy, Precision, Recall, F1 Score, Dice Score, Jaccard Index/Coefficient, AUROC, AUPRC etc.
- Draw figures of our runs
- Train and save models

- Change the dataset to some other huge dataset (for example a dataset is available for on Kaggle for cars segmentation but it is 3D images dataset)
- Write final report

Dataset

I searched the web for suitable datasets and I came across the Br35H dataset on Kaggle (https://www.kaggle.com/datasets/ahmedhamada0/brain-tumor-detection). This dataset contains 800 images with training, validation, and test splits (500, 200, and 100 images respectively). This is my initial dataset and I might change this dataset later, as discussed with Yücel hoca, in order to perform testing and validation on different datasets or a different problem domain.

Code Availability

Our code is available on GitHub: https://github.com/MoaazK/comp511-project

References

Dosovitskiy, Alexey, et al. “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale.” *arXiv*, 2020. *arXiv*, https://arxiv.org/abs/2010.11929.

Oktay, Ozan, et al. “Attention U-Net: Learning Where to Look for the Pancreas.” arXiv, 2018. arXiv, https://arxiv.org/abs/1804.03999.

Petit, Olivier, et al. “U-Net Transformer: Self and Cross Attention for Medical Image Segmentation.” *arXiv*, 2021. *arXiv*, https://arxiv.org/abs/2103.06104.

Ronneberger, Olaf, et al. “U-Net: Convolutional Networks for Biomedical Image Segmentation.” *LNCS*, vol. 9351, 2015, pp. 234-241, https://doi.org/10.1007/978-3-319-24574-4_28.

Vaswani, Ashish, et al. “Attention is All you Need.” *Advances in Neural Information Processing Systems*, vol. 30, 2017. *NeurIPS*, https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.