Using GAN synthesized images of MRI brain scans for improved training of U-Net brain tissue segmenters

Russel Arbore

Introduction

MRI brain scans are segmented by pathologists and radiologists in order to determine what regions of a patient's brain are tumor and non-tumor tissue. As this process is manual, it is invariably error prone. An alternate approach to this task is to analyze a given image mathematically. Segmentation is a type of computer vision problem involving the determination of regional classifications for an image [1], and thus Convolutional Neural Networks, or CNNs, can be applied. A special variant of CNNs called U-Net has been shown to perform better than standard CNNs on segmentation tasks [2]. However, machine learning methods only work when a reasonable amount of training data is available. Often medical datasets are limited in size due to the impracticality, legalities, and ethics of compiling large amounts of patient data into one place. I propose a new approach which uses Generative Adversarial Networks (GANs) [3] to generate training data to improve the performance of U-Nets on small datasets.

Definitions

- **Convolutional Neural Network (CNN)** - A class of neural networks that uses convolutional layers acting on image inputs which may have any dimensionalities.
- **U-Net** - A type of CNN that concatenates early latent representations to future representations.
- **Generative Adversarial Network (GAN)** - A class of machine learning systems used for generating data. A GAN consists of a discriminator and a generator network. The discriminator trains to classify images from a "real" dataset as real, and generated data as fake. The generator trains to create images that "fool" the discriminator into giving real classifications.
- **DCGAN** - A class of GANs that uses convolutional layers and transposed convolutional layers for processing. This class is better suited for generating images than the traditional GAN.

Dataset & Pre-processing

- I used the MICCAI & BRATS 2019 dataset [4].
- Originally 335 subjects with 4 types of scans and 1 truth segmentation map per subject (335 subjects is very little)
- Only the FLAIR scan type was used for input
- Total dimensionality was 335x5x155x240x240, where the 1st dimension is the subject number, the 2nd is the image type (type of scan or truth segmentation), and the last 3 are the image dimensions
- Bounding box was determined containing all non-black pixels for each image
- This was done to reduce each image's size from 155x240x240 to 149x173x192
- Finally, each image was down sampled (smooth) to 64x64x64

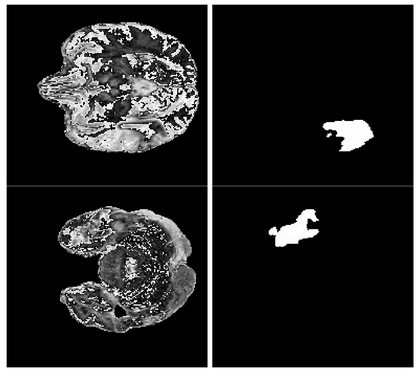

Fig. 1: Example slices from the MICCAI & BRATS 2019 dataset before down sampling. These images are each slices from 3d images. The images on the left are MRI scans, and the images on the right are the corresponding truth segmentation maps.

Final Models

U-Net

- Down sampling consisted of blocks of 2 convolutional layers followed by 2x2x2 average pools
- 4 down sampling blocks were used
- A pair of convolutional layers were used in between down sampling and up sampling
- Up sampling consisted of convolutional transpose layers followed by concatenations
- Concatenations were in between the most recent representation and the representation from the down sampling portion with the same dimensionality
- Concatenated representations were then passed through 2 convolutional layers
- 4 up sampling blocks were used to achieve original dimensionality
- All convolutional layers used a kernel size of 3 and padding size of 1
- All convolutional transpose layers had a kernel size of 2 and a stride of 2

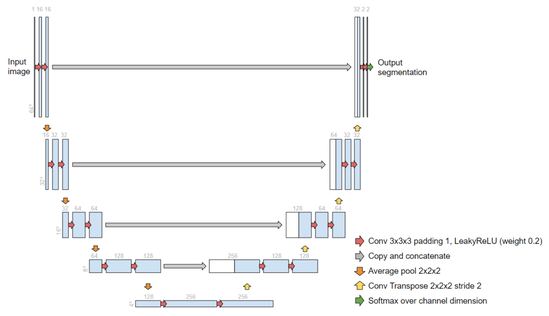

Fig. 2: The full architecture of the U-Net used for experimentation.

GAN

- Generator consisted of sections of convolutional transpose layers, batch normalization, and ReLU
- Discriminator consisted of sections of convolutional layers, batch normalization, and LeakyReLU with weight 0.2
- All convolutional layers (normal and transposed) used filter sizes of 4, padding of 2, and stride 1
- Only exceptions were the input layer of generator and output layer of discriminator who used strides of 1 and no padding
- Channel size is divided by 2 each layer in the generator and multiplied by 2 in the discriminator.
- There were 4 repeating sections in the discriminator and generator

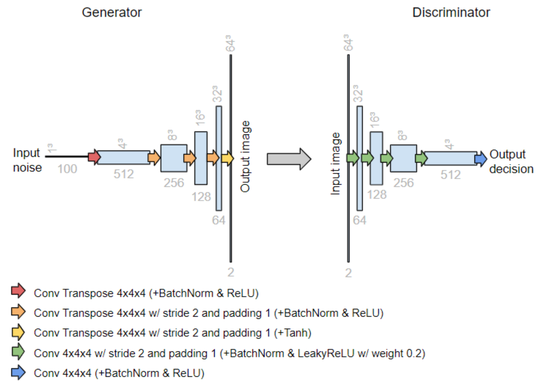

Fig. 3: The full architecture of the GAN used for experimentation.

Results

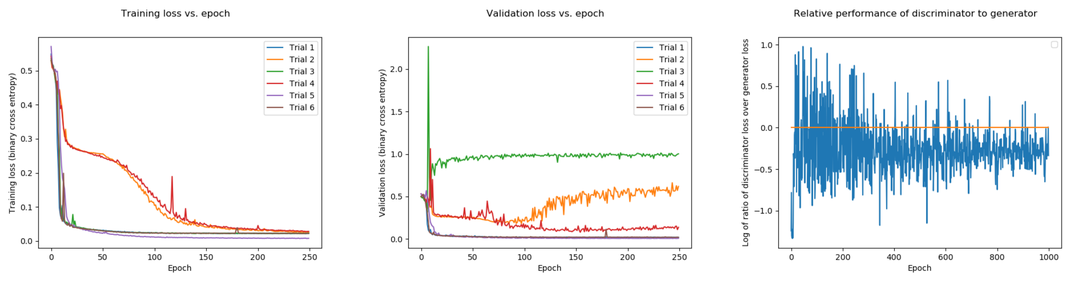

Fig. 4: From left to right, training loss of the U-Net trained for 250 epochs, validation loss of the U-Net trained for 250 epochs, log of ratio of losses of the discriminator and generator during GAN training.

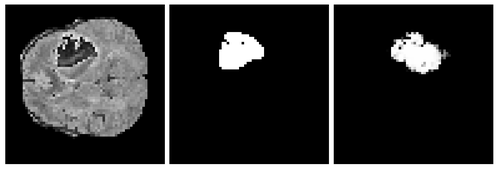

Fig. 5: Examples of trial 3's U-Net applying on real data. From left to right, the images are the flair scan, the true segmentation map, and the U-Net's prediction.

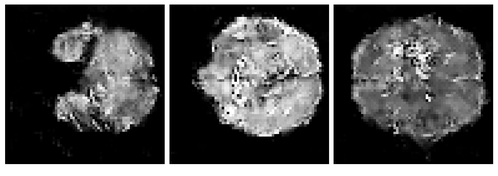

Fig. 6: Example GAN generations of scans.

Methods

- All tests were programmed in Python 3.7.6 and PyTorch
- The tests were run by an AMD Ryzen 7 2700x CPU and a EVGA RTX 2060 Super GPU
- All real data was taken from the MICCAI & BRATS 2019 dataset
- The first 300 images were used as training data and the last 35 were used for validation and testing
- All fake data was generated using random noise and a GAN trained for 1000 epochs
- All trials were run for 250 epochs with 60 batches per epoch and 35 images used for validation
- The loss function used was binary cross entropy

Trial	Images per epoch	Real/fake for training	Real/fake for validation
1	300	Fake	Fake
2	300	Real	Real
3	300	Fake	Real
4	300	Real	Fake
5	60	Fake	Fake
6	600	Fake	Fake

Conclusions

- U-Net is able to train on fake images more easily than real data as the loss decreases much faster
- Suggests images generated by GAN are less complex than real dataset
- GAN able to generate data with corresponding input and label, as U-Net can easily pick up patterns in the fake dataset in trials 1, 5, & 6
- However, since GAN images are more simple than those from the real dataset, training on GAN images currently does not result in a U-Net that works as well as one that trains on real data when validating on real data as seen in trial 3
- As seen in Fig 6, the DCGAN architecture generates rather blurry images, and the GAN loss in Fig 5 shows the discriminator outpacing the generator (discriminator consistently has lower loss after a few hundred epochs)
- U-Net trained with fake data still applies with some accuracy to real data (Fig. 5 is an example of trial 3's U-Net prediction on real data), so a likely issue is the quality of GAN generated images

Future Work

- Create a higher quality GAN
- DCGAN architecture is relatively easy to setup, but has known issues related to quality
- Certain architectures have been shown to create higher quality images, such as progressively growing GANs
- The largest bottleneck currently is the quality of GAN images, as the U-Net on its own functions correctly

Works Cited

[1] Daniel Weinland, Remi Ronfard, and Edmond Boyer. "A survey of vision-based methods for action representation, segmentation and recognition". In: *Computer Vision and Image Understanding* (Feb. 2011).

[2] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. "U-Net: Convolutional Networks for Biomedical Image Segmentation". In: *International Conference on Medical Image Computing and Computer-Assisted Intervention* (Nov. 2015).

[3] Ian Goodfellow et al. "Generative Adversarial Nets". In: *Neural Information Processing Systems Conference* (June 2014).