Generating conductivity maps from T1-MR images of human brain using 3D autoencoder like architectures

Guide: Professor Anand Joshi, Takfarinas Medani

Shreyas Dinesh Patil
Department of Electrical Engineering
Viterbi School of Engineering
University of Southern California
shreyasp@usc.edu

1 Objective:

The purpose of this research is to explore deep learning techniques for generating conductivity maps using MRI data of humans.

2 Introduction:

2.1 Imaging modalities and their uses:

Imaging Modality:

Meaning: Modality is a term used in radiology that refers a particular form of imaging. In this particular research MRI radiology modality is relevant.

1) MRI (Magnetic Resonance Imaging):
   This is an imaging technique that uses non ionizing radiation to create diagnostic images. In this procedure patient is placed in a large powerful magnet. Radio wave antenna is used to send signals into patients body and receive them back. The received signals are processed by a computer attached to the MRI scanner to generate images.

Uses of MRI:

1) Imaging of any part of the body can be obtained in any plane.
2) Uses to accurately detect abnormalities in body when other methods of diagnosis fail to provide sufficient information.

2.2 BrainSuite Software:

- BrainSuite is a collection of open source software tools that largely allows for automated processing of human MRI.
- It provides tools for visualizing and interacting with MRI data.
- Some important utilities of the software are to extract and parameterize the inner and outer surfaces of the cerebral cortex, segment and label gray and white matter structures and analyzing diffusion imaging data.

2.3 Autoencoder:

- Autoencoder are neural networks with output same as input.
- The purpose of the autoencoder is to learn a latent representation of the input which then can be used to reconstruct the input.
- Therefore autoencoders find their use in representation learning which can be further used in other tasks such as classification, detection, segmentation, etc..
- Autoencoder architecture consists of 2 parts, encoder and decoder
- The encoder transforms the input into a different representation, embedding of the input. This embedding contains only the most important information from the input which approximately represents the input.
- The decoder takes input as the output of encoder and tries to reconstruct the input from the embedding.
- Two major types of loss functions are used: mean squared error and Kullback-Leibler divergence.
- The MSE loss is between the output image and ground truth image is given by,

$$MSE = \frac{1}{NM}\sum_{i}^{N}\sum_{j}^{M}(\bar{I}_{ij} - I_{ij})^2$$

- The KL divergence is given by,

$$D_{KL}(p||q) = -\sum_{x} p(x)\log\left(\frac{q(x)}{p(x)}\right)$$

- Minimizing the KL divergence means to make the autoencoder output from a distribution that is similar to the distribution of the input image.

Types of autoencoders:

1) Sparse autoencoders: Autoencoders having sparsity penalty in their training criterion are sparse autoencoders. The sparsity penalty can be L1 regularization or KL-divergence. Activations of hidden layers are penalized using sparsity penalty so that only a few nodes are encouraged to be active in a single forward pass of the training.

2) Denoising autoencoders: Autoencoders which are trained on noisy input to regress noiseless output are denoising autoencoders. These autoencoders remove noise from the input data to learn meaningful underlying data.
3) Variational autoencoders: Variational autoencoders are generative models. This autoencoder is regularized to avoid overfitting and ensures that the embedding has good properties that enable generative process. Different from standard autoencoder this autoencoder instead of encoding an input as a single point, it encodes it as a distribution over latent space. The training is as follows:
    i) The input is encoded as distribution over latent space.
    ii) Point from the latent distribution is sampled.
    iii) The sampled point is decoded and reconstruction error is calculated.
    iv) This reconstruction error is backpropagated.

    The loss function is composed of reconstruction error on the final layer and a regularization term on the embedding layer.

3 Proposed architecture of 3D convolutional encoder-decoder architecture:

Tensor size = (128, 128, 72) – Input
Convolutional layer 1: filters = 16, kernel = (3, 3, 3)
Maxpool layer 1: pool size = (2, 2, 2), stride = (2, 2, 2)
Convolutional layer 2: filters = 32, kernel = (3, 3, 3)
Maxpool layer 2: pool size = (2, 2, 2), stride = (2, 2, 2)
Convolutional layer 3: filters = 96, kernel = (2, 2, 2)
Maxpool layer 3: pool size = (2, 2, 2), stride = (2, 2, 2)
This is the latent representation
Unpool layer 1: pool size = (2, 2, 2), stride = (2, 2, 2)
Deconvolutional layer 1, filter = 96, kernel = (2, 2, 2)
Unpool layer 2: pool size = (2, 2, 2), stride = (2, 2, 2)
Deconvolutional layer 2, filter = 32, kernel = (3, 3, 3)
Unpool layer 3: pool size = (2, 2, 2), stride = (2, 2, 2)
Deconvolutional layer 3, filter = 16, kernel = (3, 3, 3)

4 Experiments:

4.1 Data:

The dataset is a subset of the Reading Brain Project data focusing on 56 subjects. For each subject T1-weighted MRI, diffusion weighted images, b-values file, gradient files are used for this research.

The dataset can be found at **https://openneuro.org/datasets/ds002317/versions/1.0.0**

4.2 Preprocessing the data to generate Diffusion Tensor Images:

Dwi2cond function in SimNIBS software package was used to generate diffusion tensor images(DTI) from diffusion weighted images, b-values file, gradient files for each respective subject.

Note: SimNIBS is a free and open source software package for the simulation of non-invasive brain simulation.

4.3 Preprocessing of Data for alignment of images:

T1-MR images are used to generate bias-field-corrected image using the BrainSuite software. These bias-field-corrected images along with diffusion images, gradient file (contains diffusion gradients directions), b-values file (contains b-values of diffusion scan) are used to generate geometric distortion corrected and coregistered diffusion images using the BrainSuite diffusion pipeline(http://brainsuite.org/processing/diffusion/). The BDP pipeline also estimates the diffusion tensors.

BrainSuite: http://brainsuite.org/

4.4 Exploratory work done:

The T1-MR images were used as input to the model and DTI data generated using SimNIBS dwi2cond function was used as output. The data for 56 subjects had same DTI dimensions but different T1-MR image dimensions for each subject. So resizing the input images to output size was needed. But simply resizing the images to the same size as output images was not sufficient as the images needed to be coregistered i.e. aligned to each other when overlaid. The BDP pipeline was used for coregistration of diffusion weighted images.

The diffusion tensors generated using BDP pipeline were then used as output and bias field corrected diffusion images were used as input to the model.

4.5 To be done:

Currently the network is not giving tangible results. Modifying the network and generating more data can help in getting good results. Also, only 1 value per voxel is being considered in diffusion tensors generated using BrainSuite Diffusion Pipeline(BDP) while training. All values for each voxel can be included to getter better results. The exploratory work was just tested using 8 subjects data to show proof of concept. Dataset can be generated on remaining subjects using the preprocessing for data alignment suggested.

References:

[1] GitHub repo: https://github.com/shreyasdpatil/Directed-Research

[2] https://radiopaedia.org/articles/mri-2

[3] https://towardsdatascience.com/understanding-variational-autoencoders-vaes-f70510919f73