C1M3_Assignment

June 25, 2020

Image Source

1 Brain Tumor Auto-Segmentation for Magnetic Resonance Imaging (MRI)

Welcome to the final part of the "Artificial Intelligence for Medicine" course 1!

You will learn how to build a neural network to automatically segment tumor regions in brain, using MRI (Magnetic Resonance Imaging) scans.

The MRI scan is one of the most common image modalities that we encounter in the radiology field.

Other data modalities include: - Computer Tomography (CT), - Ultrasound - X-Rays.

In this assignment we will be focusing on MRIs but many of our learnings applies to other mentioned modalities as well. We'll walk you through some of the steps of training a deep learning model for segmentation.

You will learn:

- What is in an MR image
- Standard data preparation techniques for MRI datasets
- Metrics and loss functions for segmentation
- Visualizing and evaluating segmentation models

1.1 Outline

Use these links to jump to particular sections of this assignment!

  - Section **??**

- Section **??**
- Section **??**
  - Section **??**
  - Section **??**
  - Section **??**

1.2 Packages

In this assignment, we'll make use of the following packages:

- `keras` is a framework for building deep learning models.
- `keras.backend` allows us to perform math operations on tensors.
- `nibabel` will let us extract the images and labels from the files in our dataset.
- `numpy` is a library for mathematical and scientific operations.
- `pandas` is what we'll use to manipulate our data.

1.3 Import Packages

Run the next cell to import all the necessary packages, dependencies and custom util functions.

```
In [1]: import keras
        import json
        import numpy as np
        import pandas as pd
        import nibabel as nib
        import matplotlib.pyplot as plt

        from tensorflow.keras import backend as K

        import util
```

```
Using TensorFlow backend.
```

1 Dataset ## 1.1 What is an MRI?

Magnetic resonance imaging (MRI) is an advanced imaging technique that is used to observe a variety of diseases and parts of the body.

As we will see later, neural networks can analyze these images individually (as a radiologist would) or combine them into a single 3D volume to make predictions.

At a high level, MRI works by measuring the radio waves emitting by atoms subjected to a magnetic field.

In this assignment, we'll build a multi-class segmentation model. We'll identify 3 different abnormalities in each image: edemas, non-enhancing tumors, and enhancing tumors.

1.4 1.2 MRI Data Processing

We often encounter MR images in the DICOM format. - The DICOM format is the output format for most commercial MRI scanners. This type of data can be processed using the pydicom Python library.

In this assignment, we will be using the data from the Decathlon 10 Challenge. This data has been mostly pre-processed for the competition participants, however in real practice, MRI data needs to be significantly pre-preprocessed before we can use it to train our models.

1.3 Exploring the Dataset

Our dataset is stored in the NifTI-1 format and we will be using the NiBabel library to interact with the files. Each training sample is composed of two separate files:

The first file is an image file containing a 4D array of MR image in the shape of (240, 240, 155, 4). - The first 3 dimensions are the X, Y, and Z values for each point in the 3D volume, which is commonly called a voxel. - The 4th dimension is the values for 4 different sequences - 0: FLAIR: "Fluid Attenuated Inversion Recovery" (FLAIR) - 1: T1w: "T1-weighted" - 2: t1gd: "T1-weighted with gadolinium contrast enhancement" (T1-Gd) - 3: T2w: "T2-weighted"

The second file in each training example is a label file containing a 3D array with the shape of (240, 240, 155).
- The integer values in this array indicate the "label" for each voxel in the corresponding image files: - 0: background - 1: edema - 2: non-enhancing tumor - 3: enhancing tumor

We have access to a total of 484 training images which we will be splitting into a training (80%) and validation (20%) dataset.

Let's begin by looking at one single case and visualizing the data! You have access to 10 different cases via this notebook and we strongly encourage you to explore the data further on your own.

We'll use the NiBabel library to load the image and label for a case. The function is shown below to give you a sense of how it works.

```
In [2]: # set home directory and data directory
        HOME_DIR = "./BraTS-Data/"
        DATA_DIR = HOME_DIR

        def load_case(image_nifty_file, label_nifty_file):
            # load the image and label file, get the image content and return a numpy array fo
            image = np.array(nib.load(image_nifty_file).get_fdata())
            label = np.array(nib.load(label_nifty_file).get_fdata())

            return image, label
```

We'll now visualize an example. For this, we use a pre-defined function we have written in the `util.py` file that uses `matplotlib` to generate a summary of the image.

The colors correspond to each class. - Red is edema - Green is a non-enhancing tumor - Blue is an enhancing tumor.

Do feel free to look at this function at your own time to understand how this is achieved.

```
In [3]: image, label = load_case(DATA_DIR + "imagesTr/BRATS_003.nii.gz", DATA_DIR + "labelsTr/
        image = util.get_labeled_image(image, label)

        util.plot_image_grid(image)
```