Imperial College London

Derpartment of Mathematics

Segmentation of CT scans into Atrium/non Atrium

Thomas Vogel

CID:

Supervised by Prof Giovanni Montana

14th August 2015

Submitted in partial fulfilment of the requirements for the MSc in Statistics of Imperial College London

Abstract

This is a LaTeX template to be used for project theses by the students of the Imperial College MSc in Statistics. Please have a look at the `*.tex` source and the code comments therein. You may use this template with only minor changes, or make any major style changes you desire (e.g. redesigning the title page, adding headers,...), or you may use a different template, or you may write your own approach from scratch, or you may just use some bits and pieces from the template's LaTeX code. It's up to you. Note that resources on how to use LaTeX are available on Blackboard under 'R&LaTeX intro'.

This template will quote relevant sections from the MSc in Statistics student handbook throughout; e.g.

> " The abstract should be a brief statement of the aims and outcomes of the project, to summarise/advise even for a casual reader!"

Acknowledgements

Thank you supervisor/friends/family/pet.

“ Include an acknowledgement.”

Table of Contents

1 Background Material: on convolutional neural networks **6**

1.1 Convolutional Layers 6

1.1.1 Convolution 6

1.1.2 Pooling Layer 7

1.1.3 Typical Architecture 7

2 Setting up the problem **9**

2.1 Data 9

2.2 The Tri-Planar Method 9

2.3 Technological Details 9

2.3.1 Libraries 9

2.3.2 Computer Power 9

1 Background Material: on convolutional neural networks

We first start with a brief explanation of Convolutional Neural Networks (CNNs), the kind of neural network which will be relied on for our architecture to classify the voxels into either being in the atrium or not. CNNs are a special type of Feed-Forward Neural Networks that are particularly effective for input types that a grid-like structure such as an image, a video or a time-series where the relevant information is encoded locally. CNN have recently been responsible for some of the state-of-the-art results results in image recognition [references], speech recognition [references], and something else [reference].

1.1 Convolutional Layers

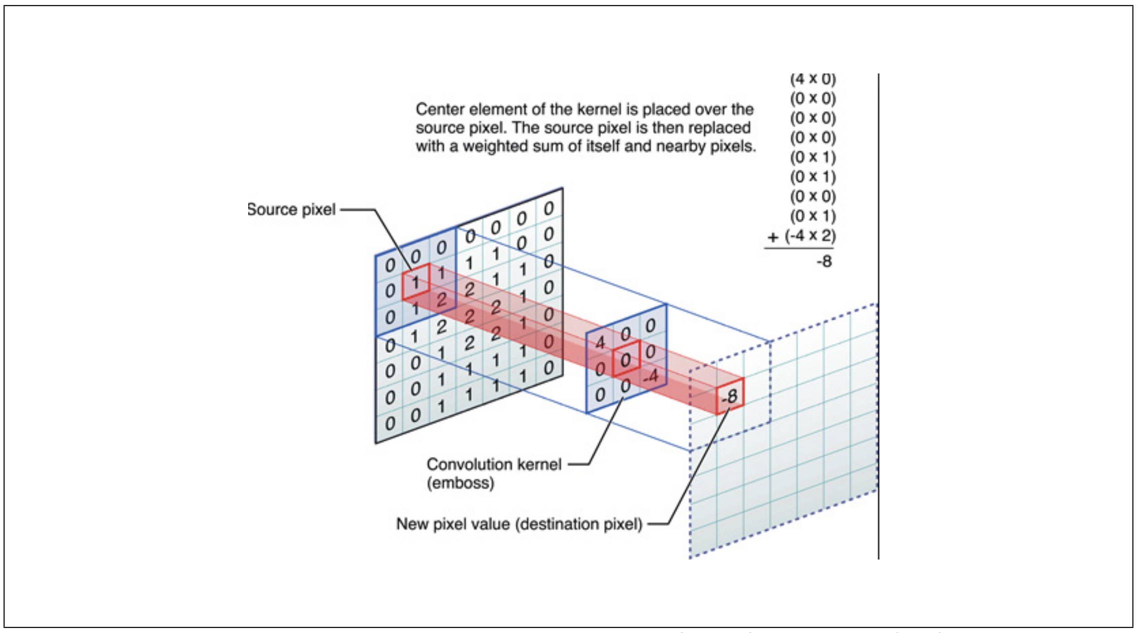
A convolutional neural network (CNN) is a specialised kind of feed-forward Neural Network where the first few layers of the architecture are convolutional layers. A convolutional layer is typically composed of two sub-layers: a convolution layer, and a subsampling layer.

1.1.1 Convolution

The convolution layer is responsible for implementing a convolution. As opposed to a fully connected layer where all the input nodes at a given layer are connected to a given output node in the next layer, a convolution restricts the number of relevant nodes to a small local subset of input nodes. Additionally the same convolution operation, i.e. the same set of weights or kernel, is applied to the possible subset of input nodes. We say that the weights are shared across the inputs nodes and this greatly reduces the number of parameters to train. The set of output nodes for one convolution layer is called a feature map, and, in the case of a two dimensional image, is a kind of slightly reduced two-dimensional image which detects the presence of a particular feature. Having only one feature map isn't necessarily useful, but feature maps can be stacked in parallel to produce a set of feature maps, each responsible for detecting a particular type of feature which might be present in the image.

The benefits of this approach are three folds :

- sparse interaction: every output node is connected to a local subset of inputs instead of to all of of them.
- parameter sharing: the same kernel is used for every output node in a given feature map.


Center element of the kernel is placed over the source pixel. The source pixel is then replaced with a weighted sum of itself and nearby pixels.
Source pixel
Convolution kernel (emboss)
New pixel value (destination pixel)
(4 x 0)
(0 x 0)
(0 x 0)
(0 x 0)
(0 x 1)
(0 x 1)
(0 x 0)
(0 x 1)
+ (-4 x 2)
-8
Figure 1.1: The convolution layer. Illustration of the formation of a feature map.

- translational equivariance: shifting the input results in an equivalent shift in the output.

Every node in the feature map is then passed to an activation unit which non-linearly transforms the output from that node, thereby introducing some non-linearity into the system. These activation units usually consist of either a Tanh unit or a Rectified Linear Unit (ReLU)

1.1.2 Pooling Layer

The pooling layer reduces the output with a local summary statistic such as the maximum or the average. This reduces the layer size and adds local translational invariance.

1.1.3 Typical Architecture

A typical convolutional neural network architecture consists of an input layer, a number of convolutional layers, a number of fully connected layers, followed by an output layer. Each layer in turn are combinations of the nodes of the previous layer thus representing more abstract features the deeper its location in the architecture.

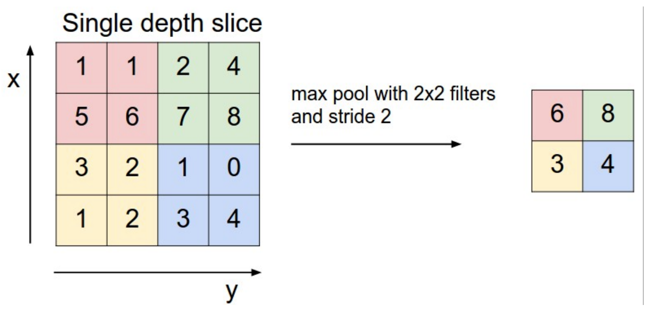


Figure 1.2: The convolution layer. Illustration of the formation of a feature map.

2 Setting up the problem

2.1 Data

The data used is a set of 27 3D CT images that are 480*480*50 in dimension. They come in 2D arrays in DICOM files. The labelling of all the voxels come in a 3D matrix in a NRRD file.

2.2 The Tri-Planar Method

Classifying the voxels require building an input set containing local and global information to allow the Neural Network to extract relevant information and aid in its choice. One of the cheap ways of doing so is to use the so-called tri-planar method. This consists in generating 3 perpendicular square patches of a given size for each voxel to classify. Each patch is then fed into a different input channel of the Convolutional Neural Network, and are then connected to the classifying layer using a fully connected layer.

2.3 Technological Details

2.3.1 Libraries

We use an open-source library called Torch in Lua to train the Neural Networks and Python and a number of its libraries to handle all the logistics from generating the datasets to producing plots of segmentation results.

2.3.2 Computer Power

The code for training Neural Networks has been written to work on multi-GPU platforms that increase the speed of training by around 100 times. The graphic cards consist in two NVIDIA Tesla K40m and two Tesla K20Xm.