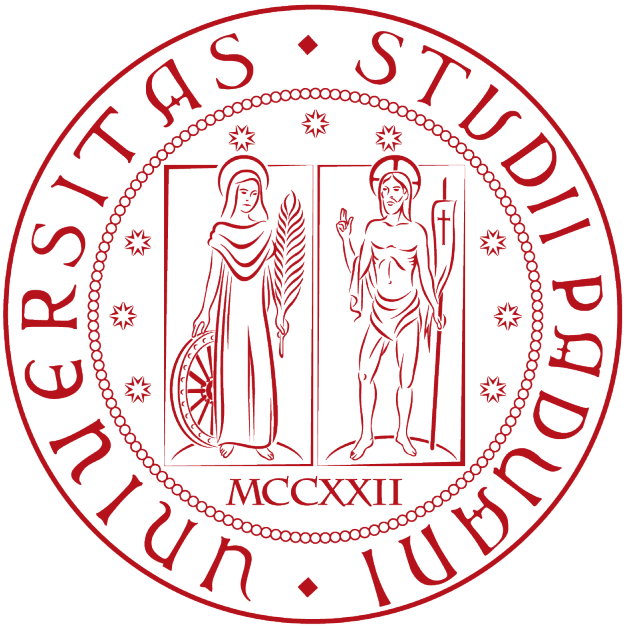
University of Padova

Bachelor of Science in Computer Engineering



Early diagnosis of Alzheimer with DeepLearning

Student:
Ivancich Stefano

Supervisor:
Prof. Nanni Loris

Academic Year 2018-2019
15 July 2019

This humble work is dedicated to Elon Musk, Andrew Ng, Ray Dalio, Chamath Palihapitiya, Charlie Munger, Jordan Peterson and Yuval Noah Harari for mentoring me through their books, courses, teachings, and online contents. This allowed me to see my life as a machine and work assiduously to maintain and improve it. Not doing this randomly. But doing it systematically, always keeping the cause-and-effect relationships in mind. And giving me a far deeper understanding of the human condition, the state of the world, how to think rationally, and, most importantly, how to better lead a life of integrity, happiness, and kindness.

Abstract

The early diagnosis of Alzheimer's Disease (AD) and its prodromal form, Mild Cognitive Impairment (MCI), has been the subject of extensive research in recent years. Some recent studies have shown promising results in the diagnosis of AD and MCI using structural Magnetic Resonance Imaging (MRI) scans. In this paper, we propose the use of a Convolutional Neural Network (CNN) in the detection of AD and MCI. In particular, we used the 27-layered AlexNet for the binary classification on the Alzheimer's Disease Neuroimaging Initiative (ADNI) dataset in a such particular way that allow us to achieve an overall accuracy up to 75% and outperform several classifiers from other studies.
You can find and run the commented code and check new updates on the project at `https://github.com/ivaste/AlzheimerPrediction`

Contents

Chapter 1

Introduction

1.1 Alzheimer's disease

Alzheimer's disease (AD) is a neurological pathology that affects more than 47 million people worldwide, being the first cause of neurodegenerative dementia. Its prevalence is estimated to be around 5% after 65 years old and a staggering 30% for the more than 85 years old in developed countries [1], from now to 2050 it is estimated that 0.64 Billion people in the world will be diagnosed with AD. The most common symptom pattern of the disease begins with the gradual worsening of the ability to remember new information. It has become a major social and economic issue and its effects are devastating not only for the diseased but also for their families. For effective treatments to be administered that are capable to slow down the progression of the disease, an early and definite diagnosis of AD is necessary. However, the goal of reaching an early and accurate diagnosis requires an investigation of the symptomatic pre-dementia stage of the disease, called **Mild Cognitive Impairment (MCI)**, that is a condition in which an individual's thinking ability shows some mild changes that can be easily noticed by the people who are close to the affected person. This stage involves the challenging question of predicting whether MCI will (MCIc) or will not (MCInc) convert to AD.
Clinical individual diagnosis of AD is still primarily based on the neuropsychological assessment and examinations that are administered to the patients. A definite diagnosis, however, is only possible through postmortem analyses. The clinical diagnostic criteria for AD, which were developed in the 1980s by the National Institute of Neurologic and Communicative Disorders and Stroke and the Alzheimer's Disease and Related Disorders Association (NINCDS-ADRDA), are still in progress and have seen a progressive evolution. The diagnostic process was first mainly based on the presence of a cognitive impairment [2].
Neuropathological information was then introduced, based on the presence of senile plaques and neurofibrillary tangles [3].
These criteria were recently revised by the National Institute on Aging-Alzheimer's Association (NIA-AA) workgroup, which introduced additional supportive features for the diagnosis of AD, such as: 1) neurogenetic testing, 2) measurement of

cerebrospinal fluid (CSF), 3) amyloid and tau, and 4) neuronal injury biomarkers as measured through neuroimaging techniques, i.e., Magnetic Resonance Imaging (MRI) and Positron Emission Tomography (PET). Today, the diagnostics of AD is invasive, one of the main tests is rachisynthesis, which is painful and dangerous for a patient.
The introduction of imaging techniques in the diagnostic criteria of AD, and in particular of MRI and PET, was due to their ability to provide measurements of atrophy and of metabolism/amyloid markers, respectively. Changes in these features can be detected even before dementia is apparent [4, 5] thus representing a strong support for the diagnosis of AD. Besides this aspect, MRI shows the further advantage of being a non-invasive technique. Because of these reasons, in the last few years a considerable research effort has been focused on implementing and developing advanced MRI processing techniques that utilize machine learning (ML) systems for improving the diagnostic accuracy of AD. Automatic systems capable of distinguishing pathological subjects from normal subjects based on their MRI brain studies (without requiring a priori hypotheses regarding where information relevant to diagnosis is located in the images) would make significant strides towards the goal of early AD detection.

Since recently, due to the increased computational power of general purpose computers with GPU, the winning classification model represent Deep Neural Networks, and specifically Convolutional Neural Networks (CNN) in what concerns image classification. Our approach consist in solving three binary classification problems: CN vs AD, CN vs MCIc and MCInc vs MCIc.

1.2 ADNI Dataset

Data used in the preparation of this study were obtained from the Alzheimers Disease Neuroimaging Initiative (ADNI) database (http://adni.loni.usc.edu). The ADNI was launched in 2003 as a public-private partnership, led by Principal Investigator Michael W. Weiner, MD. The primary goal of ADNI has been to test whether serial magnetic resonance imaging (MRI), positron emission tomography (PET), other biological markers, and clinical and neuropsychological assessment can be combined to measure the progression of mild cognitive impairment (MCI) and early Alzheimers disease (AD). For up-to-date information, see www.adni-info.org.

1.3 DeepLearning

Nowadays, deep learning is becoming a leading machine-learning tool in the general imaging and computer vision domains [6–9].
In particular, convolutional neural networks (CNNs) have presented outstanding effectiveness on medical image computing problems and made a lot of improvement for computer-aided detection. [10, 11] employed convolutional neural

networks to improve three existing CAD systems for the recognition of colonic polyps on CT colonography, sclerotic spine metastases on body CT and enlarged lymph nodes on body CT . [12] used 3D CNN and weighted MRI scans to detect cerebral microbleeds. They address developed predictions with their 3D CNN compared to various classical and 2D CNN approaches. [13] employed CNNs to detect nuclei in histopathological images. In their work, they take small patches as input and model the output as a high peak in the vicinity of the center of each nucleus and flat elsewhere. [14] employed CNNs to detect patterns of interstitial lung diseases from 2D patches of chest CT scans. Their results show that CNNs can outperform existing methods that use hand-crafted features. However, there is a big challenge for applying deep learning into medical images. The deep learning can only benefit from large amounts of training data while it is difficult in medical area. The reasons are mainly due to high cost and privacy issues [15].

Chapter 2

Deep Learning Tools

In this section the reader will find a review of basic Neural Networks and the Backpropagation algorithm (which can be seen as the fundamental basics of Deep Learning), followed by a more detailed explanation of Convolutional Neural Network and some very important techniques like the Adam Optimizer, Regularization and Dropout. These concepts are used to create AlexNet and it is important to understand them before entering in the details of the architecture. Before we start, we want to remark that there is no real theory behind deep learning. In fact many techniques are based on intuitions or empirical results. The reader shouldn't be surprised if some concepts lack of mathematical motivations.

2.1 Neural Networks and the Backpropagation Algorithm

Artificial Neural Networks [16] are a computing model introduced in 1943, inspired by the structure of the biological neural networks that constitute our brain. A neural network is composed by a number of neurons (simple nodes that perform basic computations) that are connected to each other. In a more formal way, a neural network can be described as a directed graph whose nodes correspond to neurons and edges (also called arcs) correspond to links between them. Each neuron receives as input a weighted sum of the outputs of the neurons connected to its incoming edges. There are several types of neural networks, but we are interested in feedforward networks (in which the graph does not contain cycles). Usually we organize the network in layers: each layer contains a certain number of neurons and the inputs of the neurons at layer $l + 1$ are the outputs of layer l. Each layer has also a constant node and the weight of the arc that connects this node to the nodes of the successive layer is called bias, while the weights of the arcs that connect the other non-constant nodes to the nodes of the successive layer are simply called weights. The first layer is called the input layer and the neurons in this layer don't have any incoming edges (in this case the neurons contain the values that are given as input to the network); the last is the output layer and all the layers in between are called hidden layers. The data flows from

the input to the output layer.

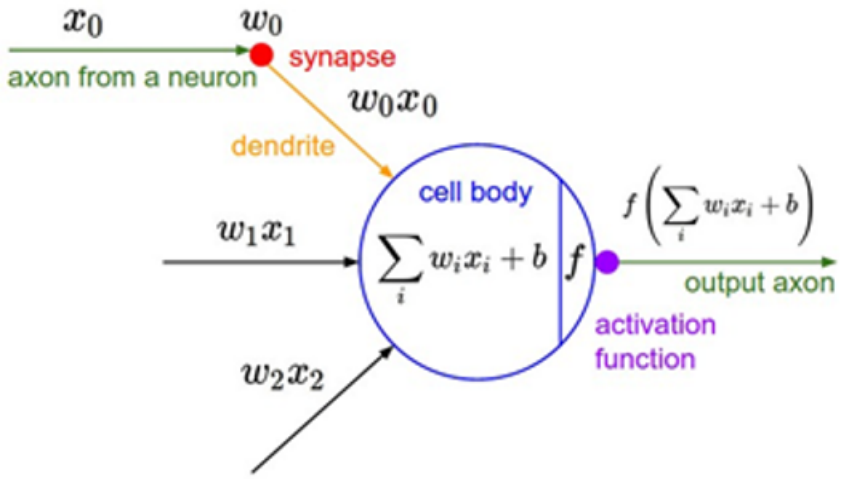


Figure 2.1: Basic Neuron.

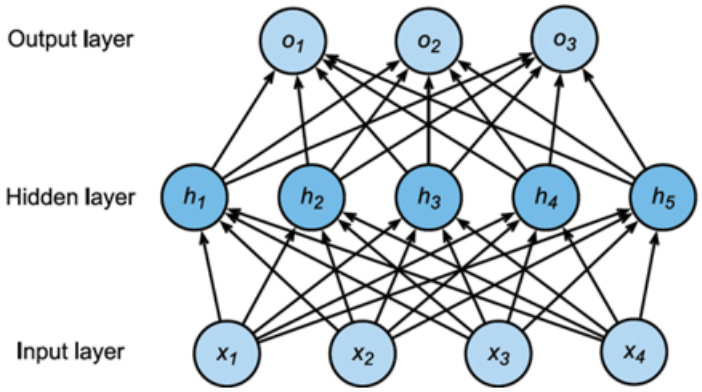


Figure 2.2: Deep Neural Network.

Single input example: $\mathbf{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}$

m examples put in an array $\mathbf{X} = \begin{bmatrix} \vdots & & \vdots \\ \mathbf{x^{(1)}} & \cdots & \mathbf{x^{(m)}} \\ \vdots & & \vdots \end{bmatrix}$

Output of a single example $y \in \{0, 1\}$

m examples in output $\mathbf{Y} = \begin{bmatrix} y^{(1)} & \cdots & y^{(m)} \end{bmatrix}$

Loss function applied to a single example $\mathcal{L}(\hat{y}, y) = -(y \log \hat{y} + (1-y) \log(1-\hat{y}))$

Cost function is the average of the Loss Function on the entire dataset. $J(\mathbf{w}, b) = 1/m \sum_{n=1}^{m} \mathcal{L}(\hat{y}^{(i)}, y^{(i)})$

Gradient Descent is an algorithm that serves to learn the parameters on the dataset, such that they minimize $\mathbf{w}$ and $J(\mathbf{w}, b)$

Activation functions are nonlinear functions are used because the composition of linear functions is always a linear function, and even with 1000 linear layers you would never be able to learn a nonlinear function.

	$g(z)$	$\frac{d}{dz}g(z)$
Sigmoid	$\frac{1}{1+e^{-z}}$	$g(z)\big(1-g(z)\big)$
Tanh	$\tanh z = \frac{e^{z}-e^{-z}}{e^{z}+e^{-z}}$	$1-\tanh^2 z$
ReLu	$\max(0,z)$	$\begin{cases}0, & z<0\\1, & z\geq 0\end{cases}$
Leaky ReLu	$\max(\alpha * z, z)$	$\begin{cases}\alpha, & z<0\\1, & z\geq 0\end{cases}$

2.2 Optimization algorithms

Deep Learning works very well on a large amount Data, but training on so many examples is very slow. Se we need advanced optimization methods that can speed up learning and perhaps even get us to a better final value for the cost function. Having a good optimization algorithm can be the difference between waiting days vs. just a few hours to get a good result.

2.2.1 Stochastic Gradient Descent with Momentum (sgdm)

Because mini-batch gradient descent makes a parameter update after seeing just a subset of examples, the direction of the update has some variance, and so the path taken by mini-batch gradient descent will "oscillate" toward convergence. Using momentum can reduce these oscillations. Takes into account the past gradients to smooth out the update.

2.2.2 Adam

As we saw in the previous section, the learning rate is a fundamental parameter for the training of a Deep Learning network, but it also difficult to find the "perfect" value. To alleviate the problem of finding the right learning rate, several optimization algorithms have been proposed and one of the most used (at the moment of the writing it is almost considered standard) is the Adam Optimizer [17].
Adam is an optimization algorithm that can be used to update the network weights based on the training data, just as any other stochastic gradient descent algorithm. The difference is that it changes the learning rate over time, and in particular, the rate is adapted based on the average first moment (the mean) and the average of the second moment of the gradients (the uncentered variance).
The algorithm has 3 main parameters: α (the learning rate), β_1 (the exponential

decay rate for the first moment estimates), β_2 (the exponential decay rate for the second-moment estimates). There is also a fourth parameter which is used to avoid a divide by zero error while updating the variable when the gradient is almost zero, and is usually set to be very small (e.g. 10^{-8}).
Advantages of Adam includes: 1) Relatively low memory requirements (though higher than gradient descent and gradient descent with momentum). 2) Usually works well even with little tuning of hyperparameters (except learning rate α)

2.2.3 Transfer Learning

Training a deep convolutional neural network from scratch is usually challenging owing to the limited amount of labeled medical data. A promising alternative is to fine-tune the weights of a network that was trained using a large set of labeled natural images. The use of pre-trained networks versus full training for medical images has been explored in [18].
This work considered four distinct medical imaging applications and investigated how the performance of deep CNNs (Convolutional Neural Networks) trained from scratch compared with the pre-trained CNNs fine-tuned in a layer-wise manner. Their experiments demonstrated that the use of a pretrained CNN with adequate fine-tuning performed as well as a CNN trained from scratch and were more robust to the size of training sets.
Another recent work [19] explored the utilization of deep CNNs to problems of computer aided detection (CADe) in the medical realm. The authors compared the performance of CifarNet, AlexNet, and GoogLeNet with different model training paradigms on the problems of detection or classification of lymph nodules and several types of lung diseases from CT images. They were able to augment their dataset significantly since they were using patches for training rather than full images. They concluded that transfer learning performed significantly better than training from scratch and that in most tasks GooglLeNet architecture proved superior, since a more complex network is able to better learn hidden structure from data.
Due to lack of computational power given to us as we are students, we used AlexNet.

2.3 Convolutional Neural Networks

Convolutional neural networks (CNN) has attracted a lot of attention due to its great success in image classification and analysis. The strong ability of CNN motivates us to develop a CNN-based prediction method of AD conversion.
Those are a class of feedforward artificial neural networks. They are used for processing data that can be represented with a grid-like topology. In fact, the grid topology introduces a notion of “distance” or “correlation” among points and CNNs, as we will see, with their structure are able to exploit this information. Classic examples are images, which can be represented as a matrix of pixels, and

time series data, which can be stored as a one-dimensional vector with each element representing a sample at a given time. Like traditional neural networks, they are composed by an input layer, an output layer, and several hidden layers in between. The name "convolutional" comes from the fact that particular layers of the network, known as convolutional layers, execute an operation called convolution on the data. In the next paragraphs we will see how convolutional layers work and what are the parameters that define them. We will then analyze the properties of CNNs and finally we will see other types of layers that can be found in a CNN architecture.

2D Convolutional Product: every cell of the result matrix is obtained by summing the element-wise product between the kernel matrix and the respective region of the input matrix overlapped by the kernel matrix + bias.
Given an $N \times N$ input matrix and a $F \times F$ filter, the output is a matrix with size $(N - F + 1) \times (N - F + 1)$ which is smaller than the input.

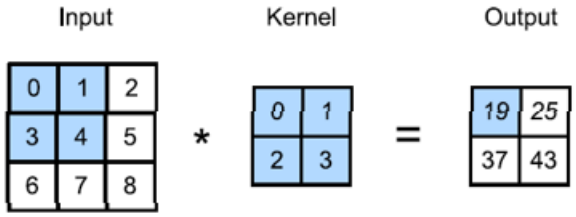


Figure 2.3: 2D Convolutional product

2.3.1 Padding

Using filters, the output picture is smaller, so having many convolution steps can reduce the picture so much to become insignificant. Moreover, in the convolutional product, the pixels along the edges are used less than the center one's. So, padding just add pixels along the edges in 3 way: 1) valid padding: not add any padding, 2) zero padding: add zeros, 3) same padding: output dimension = input dimension. $p = \frac{f-1}{2}$

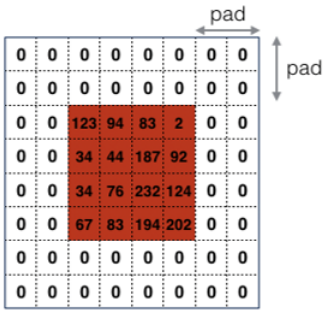


Figure 2.4: Padding.

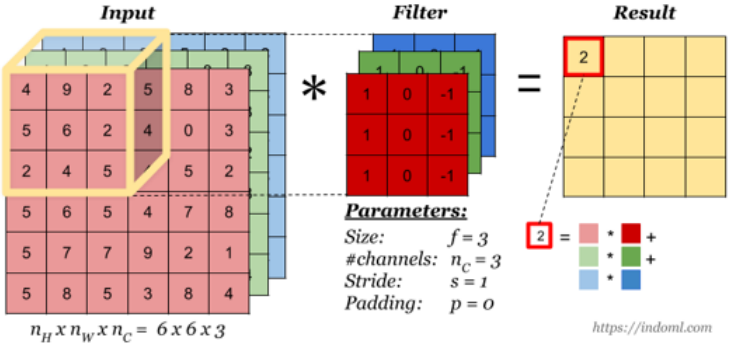
2.3.2 Stride

Do more than 1 step during convolution. The product is skipped if the filter goes outside the input. Output dimension=$\left\lfloor \frac{n+2p-f}{s} + 1 \right\rfloor$

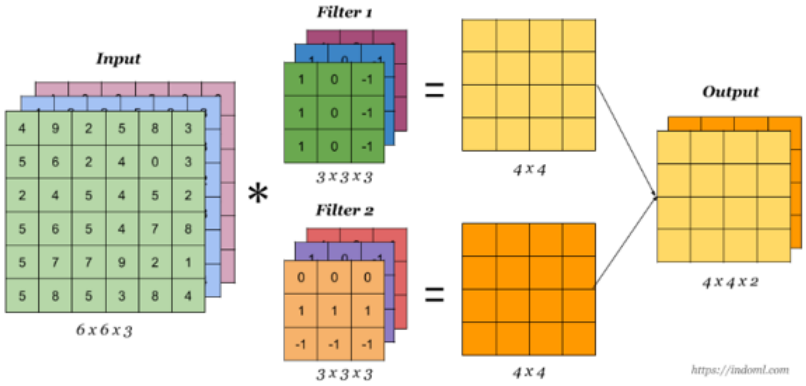
2.3.3 Convolution over volume

When the input has **more than one channel** (e.g. an RGB image), the filter should have matching number of channels. To calculate one output cell, perform

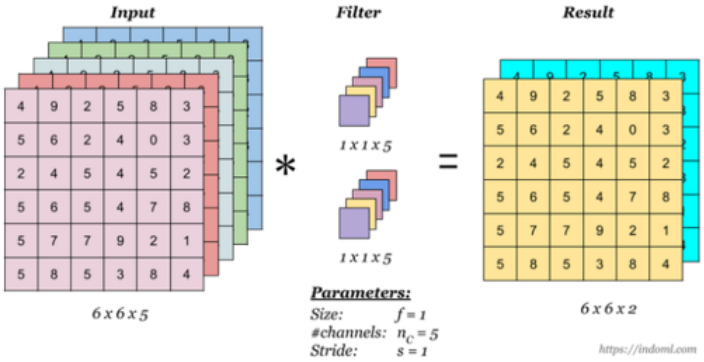
convolution on each matching channel, then add the result together.
With multiple channels:



With multiple filters: can be used in a convolution layer to detect multiple features. The output of the layer then will have the same number of channels as the number of filters in the layer.

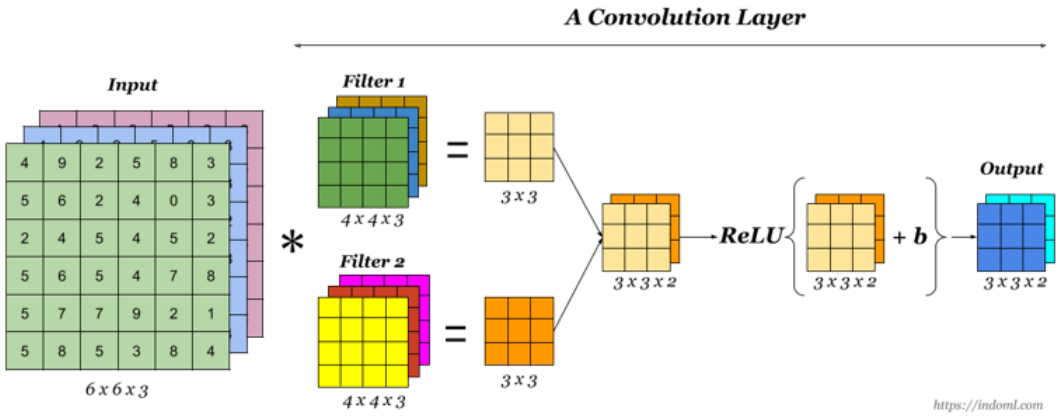


1 x 1 Convolution: the effect is to flatten or "merge" channels together, which can save computations later in the network.



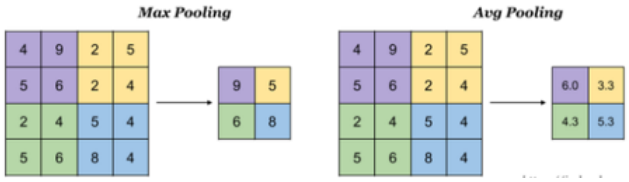
2.3.4 One convolution layer

Finally to make up a convolution layer, a bias is added and an activation function such as ReLU or tanh is applied. Bias is a vector with one parameter for each channel.
It doesn't matter how big the input is, the learnable parameters w and b depends only on the number of filters and their sizes.



2.3.5 Pooling

Pooling layer is used to reduce the size of the representations and to speed up calculations, as well as to make some of the features it detects a bit more robust.
Max pooling: like convolution but take the maximum of the region instead summing.
Average pooling: take the average.



Applied on multichannel reduce height and width but keep the number of channels n_c.
Output diemension=$\left\lfloor \frac{n_h - f}{s} + 1 \right\rfloor \times \ldots \times n_c$

2.4 Regularization

With deep learning architectures there is always a high risk of overfitting. Overfitting occurs when a model doesn't capture the underlying structure of the data, but only adapts to the training data. This means that the model becomes extremely good at predicting the data that has it has been trained on, but isn't then able to generalize and performs poorly on unseen data.

L2 regularization and Dropout are two very effective regularization techniques. [20]

2.4.1 Dropout

This technique is applied only during the training phase and it consists in randomly dropping units from the neural network. In practice, this means that we randomly set to zero a certain amount of activations in a layer. This amount is controlled by setting the probability of dropout (usually 0.5 is used).
Usually used when the training set is small. The negative side is that the cost function is not well defined anymore. Each layer has associated a probability of dropping its nodes, it eliminates some and repeat for each example. So each example has associated a different architecture that is trained. But all architecture shares weights.
The idea behind drop-out is that at each iteration, you train a different model that uses only a subset of your neurons. With dropout, your neurons thus become less sensitive to the activation of one other specific neuron, because that other neuron might be shut down at any time.

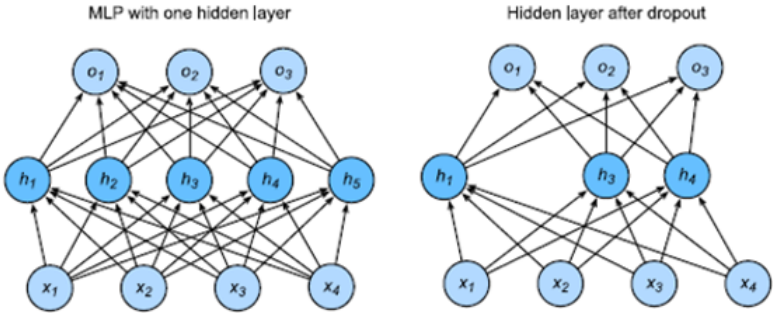


Figure 2.5: Dropout.

2.4.2 L2 Regularization

This technique consists in adding a penalty term to the loss function as the model complexity increases. In L2 regularization, also known as ridge regression, the loss function with the penalty term is defined as: $J(\mathbf{w^{[1]}}, b^{[1]}, ..., \mathbf{w^{[L]}}, b^{[L]}) = \frac{1}{m}\sum_{i=1}^{m}\mathcal{L}(\hat{y}^{(i)}, y^{(i)}) + \frac{\lambda}{2m}\sum_{l=1}^{L}\|\mathbf{w^{[l]}}\|_{\mathbf{2}}^{\mathbf{2}}$ where $\|\mathbf{w^{[l]}}\|_{\mathbf{2}}^{\mathbf{2}} = \mathbf{w^{[l]^T}}\mathbf{w^{[l]}}$
L2 regularization makes your decision boundary smoother. **If λ is too large**, it is also possible to "oversmooth", **resulting in a model with high bias.** L2-regularization relies on the assumption that a model with small weights is simpler than a model with large weights. Thus, by penalizing the square values of the weights in the cost function you drive all the weights to smaller values. It becomes too costly for the cost to have large weights! This leads to a smoother model in which the output changes more slowly as the input changes.

Chapter 3

Architecture

The architecture we propose here is based on the following **image extraction operation:** for a given voxel point, three patches of MRI 32x32 are extracted from the three planes, concatenated into a three-channel picture and resized in order to match the input size of the neural network.

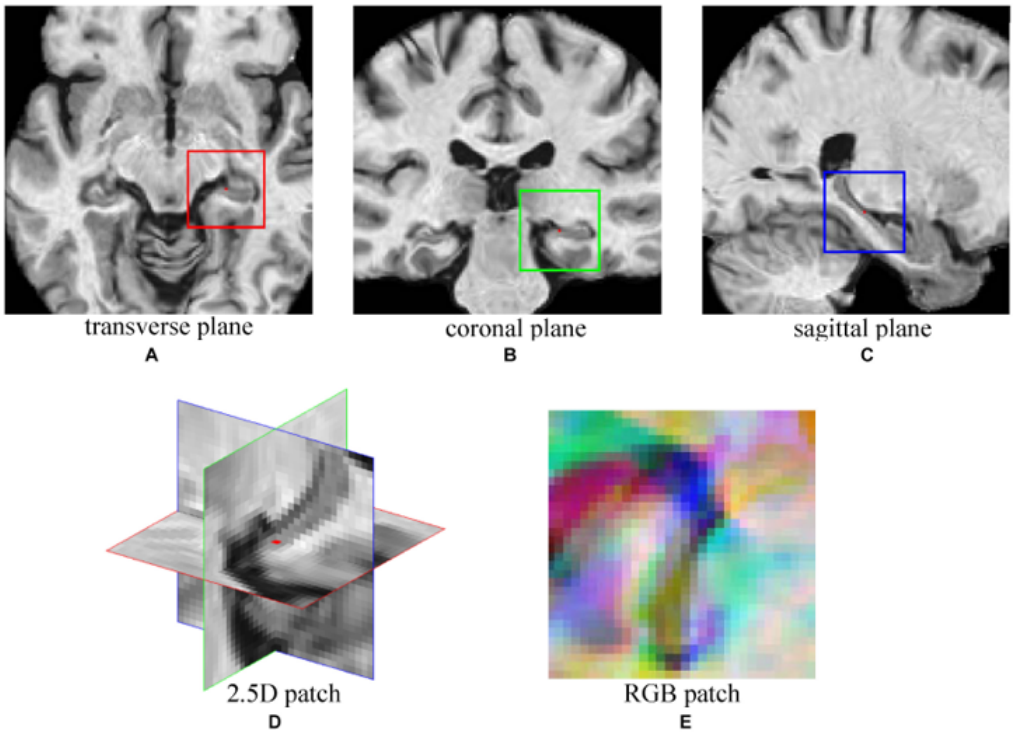


Figure 3.1: Example of image extraction operation.

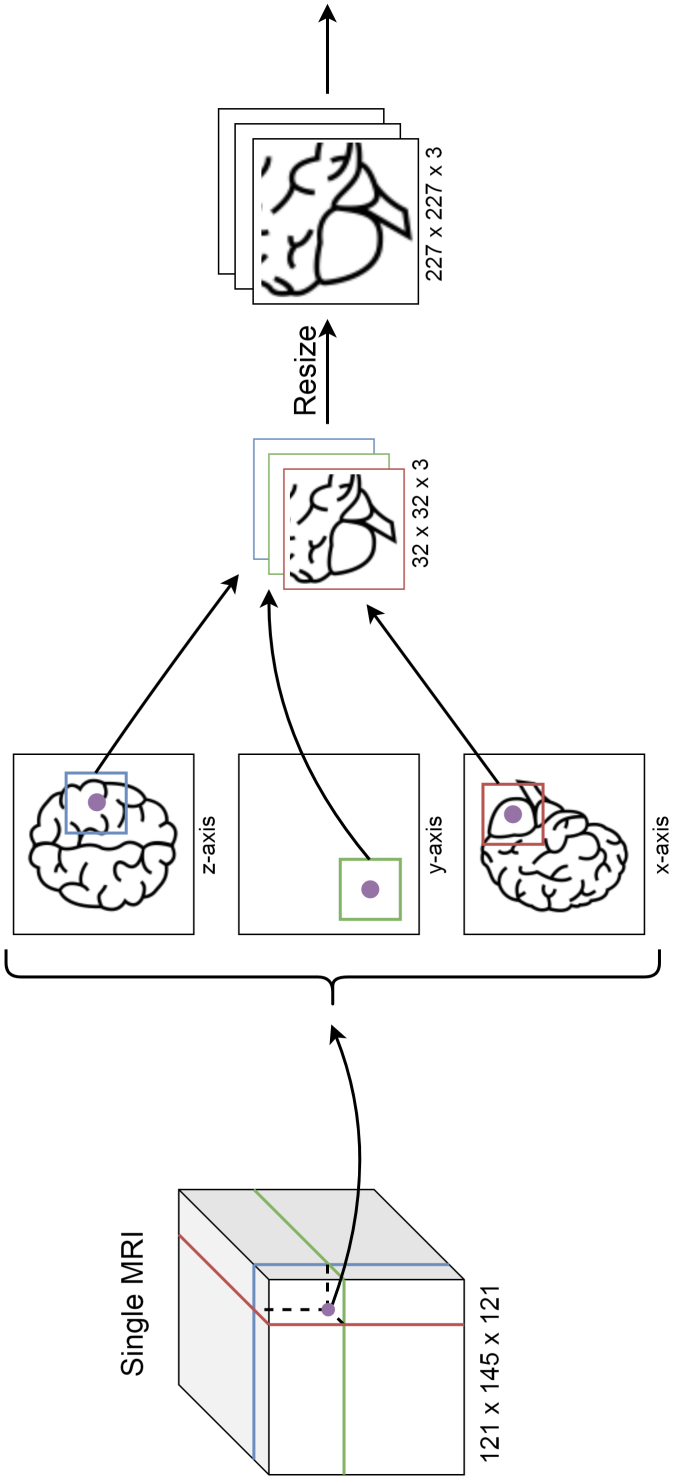


Figure 3.2: Basic image extraction operation.

The final architecture is given by extracting 100 pictures from 100 voxels of the MRI, fed each of them into our pretrained AlexNet, then combine the outputs with a weighted sum method.

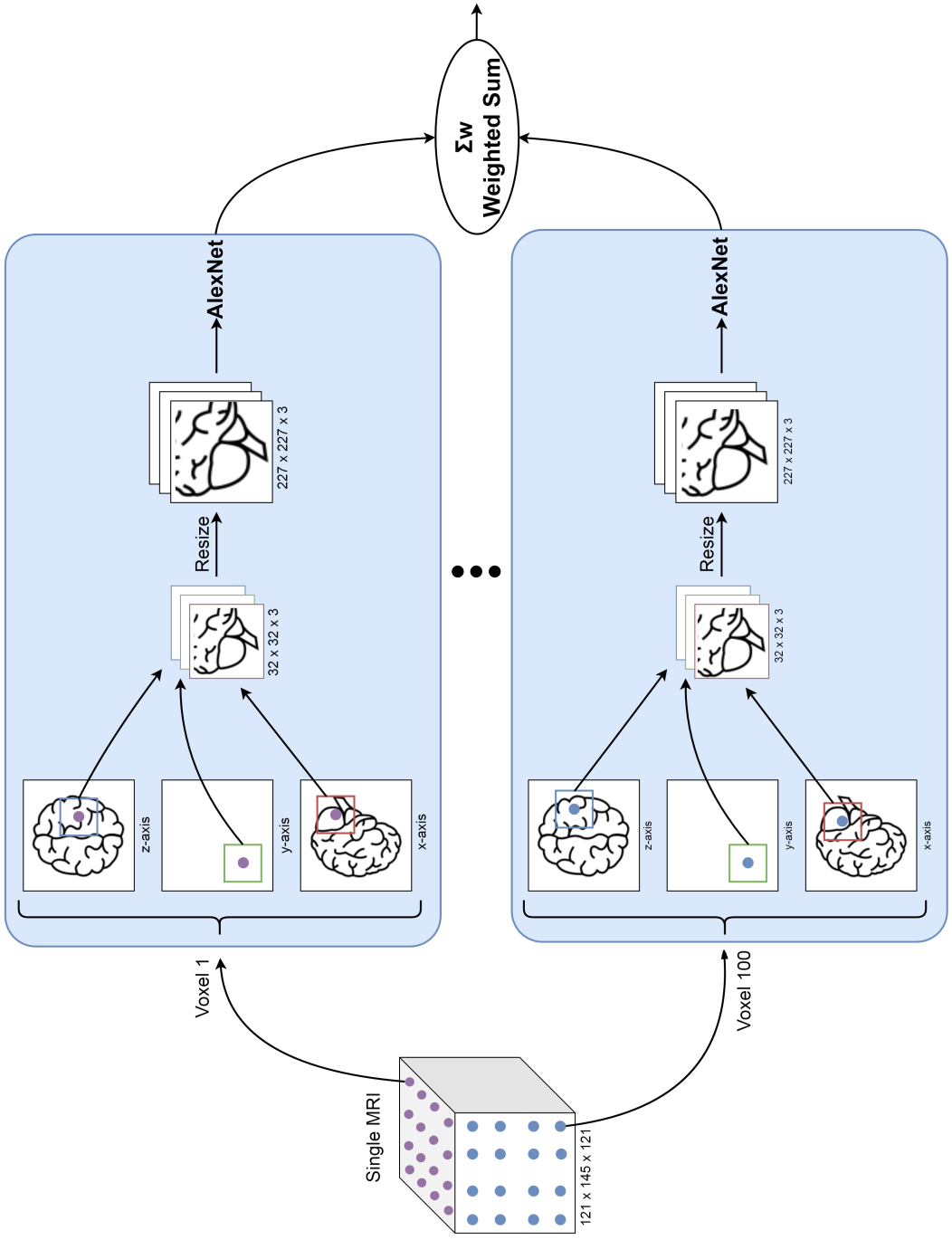


Figure 3.3: Architecture used in production.

To train the CNN AlexNet we used the following architecture, in which we extract 100 pictures from each MRI, perform some data augmentation.

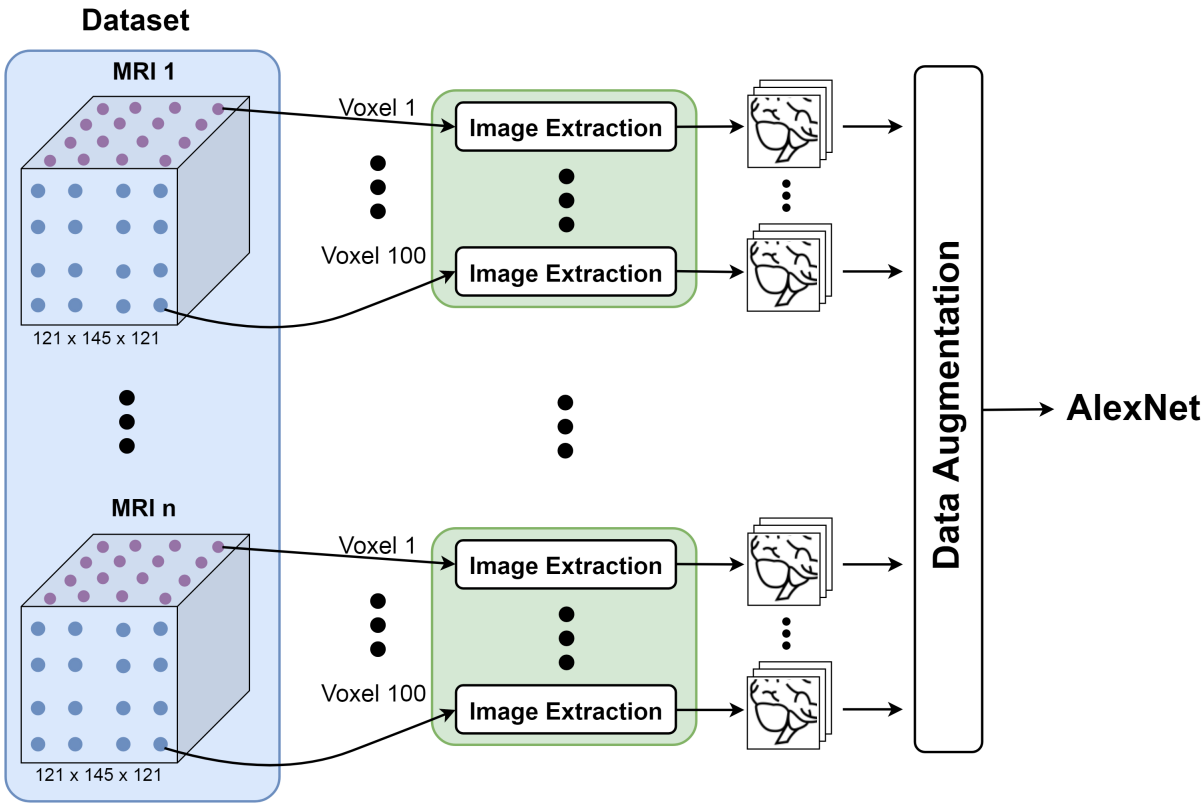


Figure 3.4: Architecture used for training.

3.1 Data Pre-Processing

The data Pre-Processing was made by the Institute of Molecular Bioimaging and Physiology, National Research Council (IBFM-CNR).
Thus, after the pre-processing we have a dataset of MRI pictures each one made of $121 \times 145 \times 121$ voxels.

3.2 Data Augmentation

A key challenge in applying deep convolutional neural networks is that sufficient training data are not always available in medical images.
One way to enlarge the dataset is to use domain-dependent **data augmentation**.
Data augmentation enables generating new training data from a smaller data set such that the new data set represents the real-world data one may see in practice.

Augmentation can extract as much information from data as possible.
In case of medical images this often comes down to mirror flipping, small-magnitude translations, weak Gaussian blurring, brightness augmentation and shadow augmentation [21].

3.3 AlexNet

As mentioned before training a deep convolutional neural network from scratch is usually challenging owing to the limited amount of labeled medical data. A promising alternative is to fine-tune the weights of a network that was trained using a large set of labeled natural images.
The use of pre-trained networks versus full training for medical images has been explored in [22]. This work considered four distinct medical imaging applications and investigated how the performance of deep CNNs (Convolutional Neural Networks) trained from scratch compared with the pre-trained CNNs fine-tuned in a layer-wise manner. Their experiments demonstrated that the use of a pretrained CNN with adequate fine-tuning performed as well as a CNN trained from scratch and were more robust to the size of training sets.
So we used AlexNet, a network that competed in the ImageNet Large Scale Visual Recognition Challenge on September 30, 2012. The network achieved a top-5 error of 15.3%, more than 10.8 percentage points lower than that of the runner up.

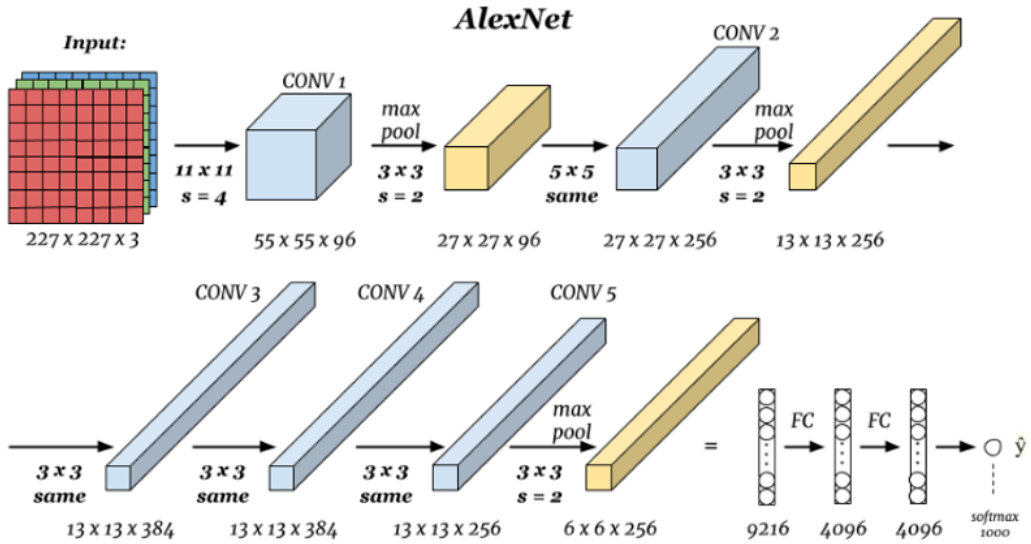


Figure 3.5: Alexnet.

Chapter 4

Experimental Tools & Results

All the code generated and methods for this thesis was implemented in MATLAB. The experiments were performed on two configurations: Intel Core i7 8750 CPU with Nvidia GeForce GTX 1060 GPU 16GB RAM and Intel Core i7-7700HQ CPU with Nvidia GeForce GTX 1050 GPU 8GB RAM.

4.1 MATLAB for DeepLearning

MATLAB (matrix laboratory) is a multi-paradigm numerical computing environment and proprietary programming language developed by MathWorks. MATLAB allows matrix manipulations, plotting of functions and data, implementation of algorithms, creation of user interfaces, and interfacing with programs written in other languages, including C, C++, C#, Java, Fortran and Python.
Although MATLAB is intended primarily for numerical computing, an optional toolbox, DeepLearning, allow working with machine learning in general but it is specifically with deep learning models.
MATLAB is available on 64-bit Linux, macOS, Windows and it can run on multiple CPUs and GPUs.

4.2 Training and Tests

Several tests have been done to evaluate the potential and the effectiveness of our model. In this section we will describe the procedure followed in each one of them. With the term hyperparameters we will refer to the parameters of the SGDM Optimizer.
The training is divided into the following steps:

1. Definition of the network (Transfer Learning from AlexNet)
2. Setting of the training parameters
3. Load the Dataset:
    - Load .mat files
    - Concatenate CN_training, CN_testing, etc... in a single cell array
    - Transform the 238 MRI images into 23800 (=238*100) 227x227x3 images (24GB)
    - Perform data augmentation to obtain a x10 larger dataset (238000 pictures, 240GB)
4. Cross Fold Validation (for 20 folds):
    - Find patterns to use for Testing (index==fold) and Training (index!=fold) and split the dataset in Training and Testing for this fold
    - Shuffle the trainingImages (and their labels, respectively)
    - Train
    - Save the trained model
    - Validate on TestSet
    - Compute Metrics

Evaluation: the proposed CNN model is evaluated by a train-validation-test scheme. The actual training of the method is carried-out on the training set, while the validation set is used for fine tuning the hyper-parameters; the overall performance of each system is assessed on the test set. The performance of the proposed model is evaluated by F-score and accuracy.

- True Positive (TP)
- True Negative (TN)
- False Positive (FP)
- False Negative (FN)

During training, for each fold, we have computed:
$Precision = \frac{TP}{TP+FP}$
$Recall = \frac{TP}{TP+FN}$
$f1 = 2 * \frac{Precision*Recall}{Precision+Recall}$
$Specificity = \frac{FP}{FP+TN}$
$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$

Using the AverageAccuracy, computed among folds, as the main metrics to drive our design.

4.3 Results

Due to the lack of memory (RAM) and computational power given to us, as we were undergraduate students, instead of converting each MRI in 100 pictures, in those tests we have extracted only 8 pictures for each MRI, trained on 3 folds instead of 20, and haven't performed any data augmentation.
Our supervisor will execute more exhaustive tests in the next weeks, to follow up the new results check `https://github.com/ivaste/AlzheimerPrediction`.

4.3.1 CN vs AD

Hyper-parameters:

- miniBatchSize = 30;
- learningRate = 1e-4;
- maxEpochs=20;
- optimizer='adam';
- "L2Regularization", 0.0001

Results:

Fold	TP	TN	FP	FN	precision	recall	f1	specificity	accuracy
1	26	56	32	6	0.4483	0.7595	0.6471	0.8125	0.6833
2	60	29	19	12	0.7595	0.833	0.7947	0.3958	0.7417
3	11	74	6	29	0.6471	0.2750	0.3860	0.0750	0.7083

Average Accuracy: 0.7111

4.3.2 CN vs MCIc

Hyper-parameters:

- miniBatchSize = 64;
- learningRate = 1e-5;
- maxEpochs=20;
- optimizer='sgdm';
- "L2Regularization", 0.0001
- "Momentum", 0.889
- 'Shuffle','every-epoch'

Results:

Fold	TP	TN	FP	FN	precision	recall	f1	specificity	accuracy
1	10	47	17	14	0.3704	0.4167	0.3922	0.2656	0.6477
2	16	52	12	16	0.5714	0.5000	0.5333	0.1875	0.7083
3	6	56	8	26	0.4286	0.1875	0.2609	0.1250	0.6458

Average Accuracy: 0.6673

4.3.3 MCInc vs MCIc

Hyper-parameters:

- miniBatchSize = 30;
- learningRate = 1e-5;
- maxEpochs=30;
- optimizer='sgdm';
- "L2Regularization", 0.0001

Results:

Fold	TP	TN	FP	FN	precision	recall	f1	specificity	accuracy
1	4	49	7	20	0.3636	0.1667	0.2286	0.1250	0.6625
2	1	46	2	31	0.3333	0.0313	0.0571	0.0417	0.5875
3	4	52	4	20	0.5000	0.1667	0.2500	0.0714	0.7000

Average Accuracy: 0.6500

Chapter 5

Conclusions

In this thesis we tackled the Early Diagnosis of Alzheimer problem with a Deep Learning approach. In particular we first analyzed the Alzheimer disease. We than introduced the theoretical elements behind the Deep Learning structures that we were going to use. Successively we described the model we have proposed, that uses convolutional neural networks (CNN), in particular transfer learning from AlexNet, that is able to automatically extract relevant features from the data.
We then presented the instruments that we used to implement our model, and the tests that we performed to evaluate it. We weren't able to obtain exceptional results due to the small computational power given to us as we are students. Nevertheless, we were still able to gain several important insights. In particular, the tests showed that our tailored model was very good at classify CN vs AD, that is an extraordinary results, because today to recognize if a person has Alzheimer different invasive medical tests must be done. With our model we need just a Magnetic Resonance.
Unfortunately CN vs MCIc and MCInc vs MCIc problems doesn't reach such extraordinary results, we think most of the problem is due to the lack of computational power.

Future Work

Given the mole of the number of parameters that constitute the model, and the limited resources (and amount of time) available, we weren't able to perform too much tests. But it would be very interesting to try to tune the different parameters and also to change the structure of the network, maybe using more sophisticated pre-trained nets like VGG-19 or Inception v4. It would be also very interesting to try some 3D convolution architecture.

Bibliography

[1] *Prévalence de la démence et de la maladie d'Alzheimer chez les personnes de 75 ans et plus: données réactualisées de la cohorte Paquid.*, Ramaroson H, Helmer C, Barberger-Gateau P, Letenneur L, Dartigues Jean- François, Rev Neurol (Paris) 159:405-411, 2003.

[2] *Clinical diagnosis of Alzheimer's disease Report of the NINCDS-ADRDA Work Group* under the auspices of Department of Health and Human Services Task Force on Alzheimer's Disease.*, McKhann, Guy, et al., Neurology 34.7 (1984): 939-939.

[3] *Editorial on consensus recommendations for the postmortem diagnosis of Alzheimer disease from the National Institute on Aging and the Reagan Institute Working Group on diagnostic criteria for the neuropathological assessment of Alzheimer disease.*, Hyman, Bradley T., and John Q. Trojanowski., Journal of Neuropathology & Experimental Neurology 56.10 (1997): 1095-1097.

[4] *The diagnosis of mild cognitive impairment due to Alzheimer's disease: Recommendations from the National Institute on Aging-Alzheimer's Association workgroups on diagnostic guidelines for Alzheimer's disease.*, Albert, Marilyn S., et al., Alzheimer's & dementia 7.3 (2011): 270-279.

[5] *Toward defining the preclinical stages of Alzheimer's disease: Recommendations from the National Institute on Aging-Alzheimer's Association workgroups on diagnostic guidelines for Alzheimer's disease.*, Sperling, Reisa A., et al. Alzheimer's & dementia 7.3 (2011): 280-292.

[6] *Online particle detection with neural networks based on topological calorimetry information.*, Ciodaro, T., Deva, D., De Seixas, J.M., et al. J. Phys. Conf. Ser. 368, 012030 (2012)

[7] *Convolutional networks can learn to generate affinity graphs for image segmentation.*, Turaga, S.C., Murray, J.F., Jain, V., et al. Neural Comput. 22, 511–538 (2010)

[8] *Efficient object localization using convolutional networks.*, Tompson, J., Goroshin, R., Jain, A., et al. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 648–656 (2015)

[9] *Deepface: closing the gap to humanlevel performance in face verification.*, Taigman, Y., Yang, M., Ranzato, M.A., et al. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1701–1708 (2014)

[10] *Guest editorial deep learning in medical imaging: overview and future promise of an exciting new technique.* Greenspan, H., van Ginneken, B., Summers, R.M. IEEE Trans. Med. Imaging 35, 1153–1159 (2016)

[11] *Improving computer-aided detection using convolutional neural networks and random view aggregation.* Roth, H., Lu, L., Liu, J., et al. IEEE Trans. Med. Imaging 35, 1170–1181 (2016)

[12] *Automatic detection of cerebral microbleeds from MR images via 3D convolutional neural networks.* Dou, Q., Chen, H., Yu, L., et al. IEEE Trans. Med. Imaging 35, 1182–1195 (2016)

[13] *Locality sensitive deep learning for detection and classification of nuclei in routine colon cancer histology images.* Sirinukunwattana, K., Raza, S.E.A., Tsang, Y.W., et al. IEEE Trans. Med. Imaging 35, 1196–1206 (2016)

[14] *Lung pattern classification for interstitial lung diseases using a deep convolutional neural network.* Anthimopoulos, M., Christodoulidis, S., Ebner, L., et al. IEEE Trans. Med. Imaging 35, 1207–1216 (2016)

[15] *Deep residual learning for image recognition.* He, K., Zhang, X., Ren, S., et al. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)

[16] *A logical calculus of the ideas immanent in nervous activity.* W. S. McCulloch and W. Pitts. The bulletin of mathematical biophysics, vol. 5, pp. 115–133, Dec 1943.

[17] *Adam: A method for stochastic optimization.* D. P. Kingma and J. Ba. CoRR, vol. abs/1412.6980, 2014.

[18] *Convolutional neural networks for medical image analysis: Full training or fine tuning?* N. Tajbakhsh, J. Y. Shin, S. R. Gurudu, et al. IEEE Transactions on Medical Imaging, 35(5):1299–1312, May 2016.

[19] *Deep convolutional neural networks for computer-aided detection: Cnn architectures, dataset characteristics and transfer learning.* H. Shin, H. Roth, M. Gao, et al. IEEE transactions on medical imaging (TMI), 35(5):1285–1298, 2016.

[20] *Dropout: A simple way to prevent neural networks from overfitting* N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Journal of Machine Learning Research, vol. 15, pp. 1929–1958, 2014.

[21] *Classification of smri for ad diagnosis with convolutional neuronal networks: A pilot study on adni.* Karim Aderghal, Manuel Boissenin, Jenny Benois-Pineau, Gwenaelle Catheline, and Karim Afdel. In International Conference on Multimedia Modeling, pages 690–701. Springer, Cham, 2017.

[22] *Convolutional neural networks for medical image analysis: Full training or fine tuning?* N. Tajbakhsh, J. Y. Shin, S. R. Gurudu, et al. IEEE Transactions on Medical Imaging, 35(5):1299–1312, May 2016.