Universitetet i Stavanger

FACULTY OF SCIENCE AND TECHNOLOGY

MASTER'S THESIS

Study programme/specialisation: Information Technology – Automation and Signal Processing	Spring semester, 2018 Open/~~Confidential~~
Author: Ulrik Moen	…… (signature of author)
Programme coordinator: Trygve Eftestøl Supervisor(s): Ketil Oppedal, Trygve Eftestøl	
Title of master's thesis: Early Detection of Alzheimer's Disease using 3D Texture Features and 3D Convolutional Neural Networks from structural MRI	
Credits: 30	
Keywords: Alzheimer's Disease, Histogram of Oriented Gradients, Local Binary Pattern, Convolutional Neural Networks	Number of pages: 45 + supplemental material/other: 7 + attached 7.zip Stavanger, 15th of June, 2018 date/year

Abstract

In 2016 there was estimated that around 47 million people in the world were living with dementia, and this number were projected to increase to 131 million by 2050. Alzheimer's disease is the most common type of dementia and the cost associated with dementia was in 2016 estimated to be around 818 billion US$ world wide. Although there does not exist any treatment that cures or significantly delay the progress of Alzheimer's disease at this time, early detection is important. It enables people with dementia and their families to be better prepared for the progression of the disease and helps with understanding the disease and its progression, which may speed up the development of a future treatment. Several different biomarkers are being explored in order to give a more reliable and earlier diagnosis. The use of Magnetic resonance imaging of the brain together with machine learning have shown promising results in discriminate between different kinds of dementia and normal controls.

In this thesis it was investigated if early diagnosis of Alzheiemr's disease could be reliably identified using magnetic resonance imaging of the brain together with either three dimensional texture features and random forest or three dimensional convolutional neural networks. The data set used consisted of 2688 MRI scans from 811 subjects, and was obtained from the Alzheimer's Disease Neuroimaging Initiative. The best accurucy results of the four-class early differential diagnosis problem was 41.1% and for the two-class problem discriminating between subjects with mild cognitive impairment who converted to Alzheimer's disease or not, was only 60.3%.

Contents

Acronyms

AD Alzheimer's disease

ADNI Alzheimer's Disease Neuroimaging Initiative

CL Convolutional Layer

CNN Convolutional neural network

CSF Cerebrospinal fluid

GM Gray Matter

HOG Histogram of oriented gradients

LBP Local binary pattern

MCI Mild cognitive impairment

MNI Montreal Neurological Institute

MRI Magnetic resonance imaging

NC Normal Control

RF Random Forest

ROI Region of interest

ReLU Rectified linear unit

TOP Three orthogonal planes

VHOG Voxel histogram of oriented gradients

WM Gray Matter

Chapter 1

Introduction

This chapter gives a general introduction to the thesis. It introduces the motivation behind the objectives to be explored, and gives a brief overview over related work, and the structure of this thesis.

1.1 Motivation

In 2016 there was estimated that around 47 million people in the world were living with dementia, and this number were projected to increase to 131 million by 2050 [1], mostly because of the general increasing age in the population. Alzheimer's disease (AD) is the most common type of dementia and the cost associated with dementia was in 2016 estimated to be around 818 billion US$ world wide [1]. Although there does not exist any treatment that cures or significantly delay the progress of AD at this time, there are treatments that may help for some of the side effects of dementia [2].

Early diagnosis of AD is important, since it enables people with dementia and their families to be better prepared for the progression of the disease. It also gives people the opportunity to make use of the available treatments that may improve some of the side affects of dementia and enhance their quality of life by making it easier to manage the disease. Long term the importance of reliable and early diagnosis of dementia might be crucial, since it might make it easier to understand the disease which may lead to future cure, and future treatment might be more efficient in the early stages.

Traditional diagnostics of dementia detects the disease late, and in order to give a reasonably accurate diagnosis a thorough examination of the persons medical history and physical and mental state is needed. Several different biomarkers are being explored in order to give a more reliable and earlier diagnosis. The use of Magnetic resonance imaging (MRI) of the brain together with machine learning have shown promising results in discriminate between different kinds of dementia and normal controls (NC) [3, 4, 5].

1.2 Thesis Objective

The main objectives in this thesis are:

1. Investigate if early detection of AD can be reliable identified using MRI of the brain and either three dimensional (3D) texture features with random forest (RF) or 3D convolutional neural networks (CNN).

2. Compare the classification results between 3D texture features with RF and 3D CNN.

3. Investigate how different prepossessing of MRI of the brain affects the classification results achieved by both 3D texture features with RF and 3D CNN.

1.3 Related Works

MRI biomarkers of dementia are a very active research field. In this section some recent published papers using MRI biomarkers to discriminate dementia will be presented.

Sorensen et al.[3] developed a algorithm that combined several MRI biomarkers to discriminate between NC, mild cognitive impairment (MCI) and AD using T1-weighted MRI. The biomarkers where volumetric and cortical measurements from several regions of interests (ROI), in addition to hippocampal texture and shape. They achieved a mean classification accuracy of 62.7% with 10-fold cross validation for the three-class problem using a data set of 689 subjects and support vector machine classifier[3]. The algorithm also won the CADDementia challenge [6], where they achieved a 63.0% classification accuracy on the "hidden" CADDementia test data.

In 2017 Oppedal et al. [4] used local binary pattern (LBP) three orthogonal planes (TOP) and white matter (WM) legions or normal appearing WM as a ROI from T1-weighted MRI to discriminate between NC, AD and Lewy body dementia (LBD) using RF. They achieved 79% accuracy on the three-class problem NC vs AD vs LBD and 97% on the two-class problem NC vs AD from a 10 folds nested cross validation, with a data set consisting of 109 subjects.

In 2014 Liu et al [7] developed a method which used neural network auto-encoders together with a soft max regression layer and 83 ROIs from MRI and Positron emission tomography(PET), to discriminate between NC, MCI who were stable (MCIs), MCI who converted to AD (MCIc) and AD. They reported an accuracy of 47.4% on the four-class problem, using a data set from Alzheimer's Disease Neuroimaging Initiative (ADNI) of 77 NC, 102 MCIs, 67 MCIc and 85 AD.

Payan and Montanta [5] proposed a method for classifying NC vs MCI vs AD using both 2D or 3D CNN with sparse auto-encoders. They achieved the best results using 3D CNN, and the reported an accuracy for three-class problem of 89.47%, NC vs AD of 95.39%, NC vs MCI of 86.84% and NC vs MCI of 92.11%.

In 2017 a challenge for early detection of AD was hosted on Kaggle [8], where 100 NC, 100 MCIs, 100 MCIc, and 100 AD subjects where included. The competition used data from the ANDI database, and the participating teams were given access to 430 features from the MRI, such as volumetric and cortical thickness from several and meta data. The subjects ids were anonymized. The 400 subjects included were split in a training set of 240, and test set 160. The diagnosis for training set subjects were available for the teams, while the test set diagnosis were hidden. The test set were also filled with 360 fake subjects. The competition lasted for 6 months and each team could daily submit a classification solution. The results for 50% of the test set, including fake data, were updated real time in a online leaderboard. The best team, Stavros Dimitriadis – Dimitris Liparas, achieved 62% classification accuracy on the 4-class problem NC vs MCI vs MCIc vs AD of the whole data set, when the fake data was removed [9]. The next 6 teams on the list achieved very similar results to eachother, about 55% [9].

1.4 Thesis Outline

Chapter 2 - Background

In this chapter the background theory used in this thesis will be presented, including a brief overview of AD and MCI.

Chapter 3 - Material and methods

This chapter will present the data material, the pre-processing methods, feature extraction methods and the classification methods used in this thesis.

Chapter 4 - Results

The results from the methods tested will be presented in this chapter.

Chapter 5 - Discussion and Conclusion

The results from the previous chapter are discussed in this chapter, and a conclusion of this thesis and recommendations for future work is given.

Appendix A - Additional Validation Results

More detailed validation results of the results presented in Chapter 4 is given.

Appendix B - Program Files

The program code used for the methods presented are listed and attached here.

Chapter 2

Background

In this chapter an overview over the background theory of the methods used in this thesis, and a brief overview over dementia, AD and MRI is presented.

2.1 Alzheimer's Disease

Dementia is a broad term of brain diseases that causes decline in a persons ability to think, remember and general behavior significant enough to interfere with daily tasks. There are several types dementia, where AD is the most common, AD accounts for about 60 - 80% of all cases of dementia [2].

AD is a degenerative neurological disease that normally worsens over time. The process of brain degeneration may be slow or fast [10]. Changes in the brain may occur before noticeable symptoms, such as the formation of amyloid plaque and neurofibraillary tangles [11].

There are no single specific test for diagnosis of AD, instead several different approaches must be conducted to help make a diagnosis. Examples of these approaches are examining the individual medial history, family history and inputs from family members regarding the individual possible changes in behavior and thinking skills [2]. The most common symptoms of AD and other types of dementia is memory loss, other symptoms includes change of mood and personality, disorientation, behavioral problems and depression. Some of this symptoms are also closely related to normal aging, which makes traditional diagnosis very difficult. Therefore analysis of biomarkers to identify AD at early stages is of major interest. Amyloid Beta, Neurodegeneraion detected by rise of cerebrospinal fluid (CSF) tau species and brain atrophy on MRI are important biomarkers [12].

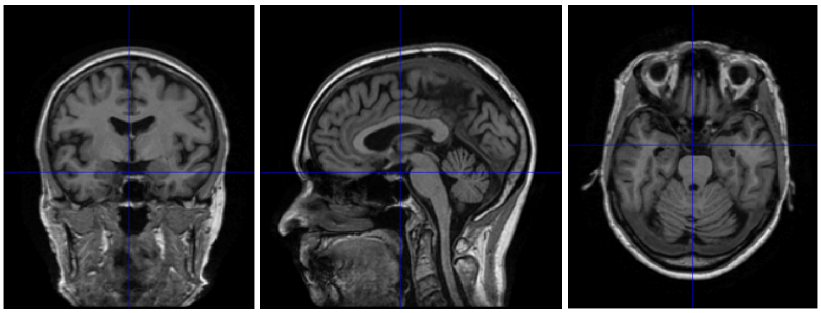
2.1.1 Mild Cognitive Impairment

People with MCI experience cognitive changes that are serious enough to be noticed by themselves and family or friends close to them, but not serious enough to be classified

as dementia, i.e. not enough to interfere with daily life [13].

MCI may be a precursor to AD. A meta-analysis of 41 studies by Mitchell and Shiri-Feshki [14] showed that the conversion rate from MCI to dementia when the the individuals were tracked for 5 years or more, averaged at 38 %.

2.2 Magnetic resonance imaging



(a) Coronal (b) Sagittal (c) Axial

Figure 2.1: T1-weighted MRI image of the three anatomical planes.

MRI is a non invasive imaging modality that produces detailed 3D anatomical images, without the damaging ionizing radiation that is found in e.g. x-ray and computed tomography. The MRI also produces excellent soft tissue contrast on tissue such as muscles, brain, fat and body fluids [15].

The basic principles of MRI is that it uses a strong magnetic field, B_0, typically between 1-3T, which will cause the nuclei to spin around an axis either parallel or anti-parallel to the direction of B_0. Although the ratio between nuclei which spins between the two directions is almost equal, the parallel direction is slightly favoured, leading to a tiny measurable net magnetization field, M_0, which is aligned parallel to B_0 [16]. A radio frequency (RF) pulse is applied perpendicular to B_0 equal to the Larmor frequency, that causes the direction of M to tilt away from B_0. The Larmor frequency is given by:

$$\omega_0 = \gamma B_0 \tag{2.1}$$

Where γ is the gyromagnetic ratio, which is a nuclei specific constant. Once the RF pulse is no longer applied, the nuclei will realign themselves to the equilibrium state, such that M is parallel to B_0 again, this is refereed to as relaxation. During relaxation, the nuclei emits their own measurable RF signal, free induction decay (FID), which can be processed to obtain a gray-scale 3D MR image.

The voxel intensities in a MR image depends on tissue in the voxel, and of the signal weighting that is used. The two main signal weighting are T1 and T2. T1 reflects the time for the displaced nuclei to return to equilibrium, while T2 reflecs the time required

for the FID value for a given tissue to decay. In this thesis T1-weighted MRI is used, and an example of T1-weighted MRI are shown in Figure 2.1, which shows the MRI from the three anatomical planes.

2.3 Local Binary Pattern

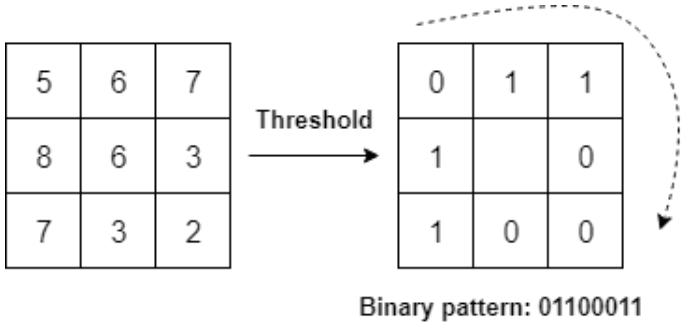


Figure 2.2: Local binary pattern.

LBP proposed by Ojala et al. in [17] provides a robust way of describing local texture patterns in an image. The original LBP operator assigns a binary label to every pixel of an image, based on the 3 by 3 neighbourhood of the centre pixel. This binary label is obtained by thresholding the local neighbourhood by the value of the centre pixel, Figure 2.2. An extension to the LBP, circular LBP [18], allows the use of variable radius and number of sampling points, R and P, Figure 2.3. The binary number for each pixel is defined in equation 2.2.

$$LBP_{P,R}(x_c, y_c) = \sum_{p=0}^{P-1} s(g_p - g_c)2^p \tag{2.2}$$

Where x_c,y_c is the position of the centre pixel, g_c and g_p is gray scale value of the centre pixel and the surrounding P pixels respectively. The thresholding function $s(x)$ is defined as:

$$s(x) = \begin{cases} 1, & \text{if } x \geq 0 \\ 0, & \text{otherwise} \end{cases} \tag{2.3}$$

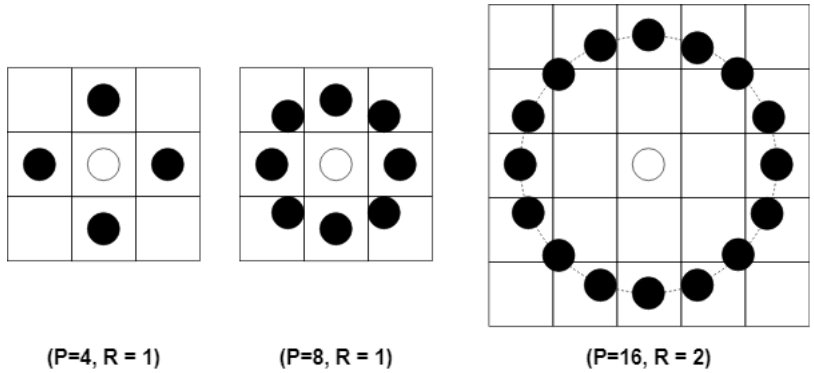


Figure 2.3: Circular local binary pattern with different radius and neighbouring points.

Rotation invariant

As the texture orientation are often arbitrary and redundant, a rotation invariant LBP pattern can be desired. The rotation invariant pattern [18] gives each pattern which are just a result of rotating another pattern the same binary label.

Rotation invariant Uniform

Pattern uniformity describes the number of transitions between 0 and 1 in the binary label. In the rotation invariant and uniform LBP pattern[18] a maximum of two transitions is allowed. All the patterns with more than two transitions are in their own bin. This reduces the number of patterns from 2^P to $P+2$ bins. [18].

2.3.1 LBP-TOP

LBP-TOP, develpoed by Zhao and Pietikäinen [19] are an extension to the LBP texture descriptor which can be used to capture local textures in a 3D image. The LBP-TOP is computed by splitting the 3D image in several slices in the XY-, XZ- and YZ-plane stacked in the last dimension. The LBP values are computed for each of these three stacks resulting in three feature histograms, and the final feature vector is obtained by e.g. concatenate these histograms together.

2.4 Histogram of oriented gradients

Histogram of oriented gradients (HOG) proposed by Dalal and Triggs [20], is a feature descriptor mostly for the purpose of object detection. HOG is computed by firstly finding the horizontal and vertical gradients in the image, usually by found by convolving the image using the 1D point discrete derivative mask for both of the horizontal and

vertical directions shown below:

$$\begin{bmatrix} -1 & 0 & 1 \end{bmatrix} and \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}$$

This gives each pixel a gradient with a direction and a magnitude, given by equation 2.4 and 2.5 respectively.

$$\theta = \arctan(\frac{g_y}{g_x}) \tag{2.4}$$

$$G = \sqrt{{g_y}^2 + {g_x}^2} \tag{2.5}$$

Where g_y is the result from the convolution in the vertical direction, and g_x is the result from the convolution in the horizontal direction. The gradient image are split into cells, e.g. $8x8$ pixels and a histogram of gradients are computed for each cell. Where each pixel gradient direction casts a vote to the histogram bin closest to the direction based on the magnitude. The number of bins in the histogram are a parameter which can be tuned. To improve the invariance to uneven illumination in a image, the cells can be organized in blocks, and normalize the cells in a block histogram, e.g. using L2-norm given by equation 2.6.

$$\mathbf{v_{norm}} = \frac{\mathbf{v}}{\sqrt{||\mathbf{v}||_2^2 + \epsilon^2}} \tag{2.6}$$

Where $\mathbf{v}$ is the non-normalized vector of all the histograms in a block, $\mathbf{v_{norm}}$ is the normalized vector, ϵ is a small positive number to avoid division by zero. The blocks are often overlapping such that each cell contributes to more than one block.

2.4.1 3D Voxel HOG

The 3D Voxel HOG (3D VHOG), proposed by Dupre and Argyriou [21] is based on the original HOG descriptor and extended for use in the third dimension by the use of voxels and 2D histograms. It works much like the original HOG except that the gradients also have a third direction, and the cells are cubic instead of square.

2.5 Random Forest

RF, proposed by Breiman[22], is an ensemble learning method. For classification multiple classification trees are constructed using bootstrap aggregation (bagging) and randomized feature subset sampling, resulting in multiple trees/ models which makes more accurate predictions than a single model alone. Each tree in the forest learns from a random subset of the training data set using a random subset of the features.

The subset of the training data is randomly sampled with replacement, leaving about one third of the training data for each tree, out of the bag (OOB) data. The OOB data is used to get a classification error rate as trees are added to the forest and can be used to measure feature importance. The classification of a test sample is done by using majority of votes from all the outputs from all the models. Some of the hyperparameters for constructing a RF is, number of trees, max number of features for each split.

2.6 Artificial Neural Networks

A Artificial Neural Network is a machine learning technique which uses many interconnected units, often referred to as artificial neurons or just neurons, to approximate a mathematical function or model. The neurons are often organized in layers, and each neuron emits an activation signal to the connected neurons in the next layer, depending on its inputs, which are loosely analogous to the human brain. These networks can be trained to solve arbitrary complex problems, such as speech recognition, object detection, medical diagnostics, etc.

2.6.1 Artificial Neuron

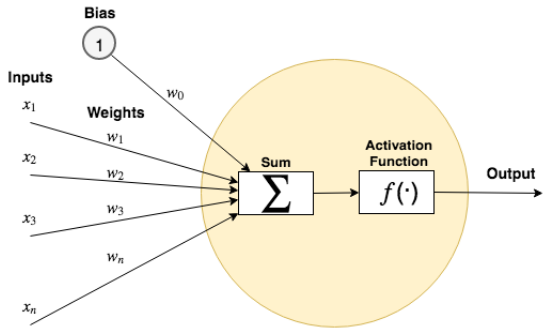


Figure 2.4: An artificial neuron with d inputs.

An artificial neuron (illustrated in Figure 2.4) is the main building block of neural networks. The neuron receives a vector of real valued inputs either from the raw inputs or from the output from other neurons in a previous layer, in addition to a bias which is usually set to 1. Each input link, including the link from the bias, to the neuron has it own modifiable weight associated with it. The weighted sums of the inputs are computed inside the neuron, often annotated net, as given by eq. 2.7 [23].

$$net_j = \sum_{i=1}^{d} x_i w_{ji} + w_{j0} = \sum_{i=0}^{d} x_i w_{ji} = \mathbf{w}_k^t \mathbf{x} \tag{2.7}$$

Where d is the number of inputs to the neuron j, x_i is the input to the neuron and w_{ji} denotes the weight in the input - output link, w_{j0} is the bias weight and the bias is set to 1. The net value, also sometimes refereed to as the preactivation value, is often used in combination with an activation function, $f(\cdot)$ (further detailed in Section 2.6.2), to compute the output value from the neuron:

$$y_j = f(net_j) \tag{2.8}$$

Where y_j is the output from neuron j. If the neuron is not in the final layer, its output is the input of neurons in the next layer.

2.6.2 Activation functions

The activation function is important, since it introduces non-linearity to the network, and it is what defines the output of a neuron given its inputs and weights. There is several activation functions available, e.g. tanh, sigmoid, etc, but the perhaps most popular activation function in the hidden layers is the rectified linear unit (ReLU), given by equation 2.9.

$$f(net_j) = max(0, net_j) \tag{2.9}$$

For classification problems the softmax activation function is often used in the final output layer. It gives each element in the output vector a value between 0 and 1, and the total sum of the elements is equal to 1. The element values can after the activation function be interpreted as probabilistic values that can tell the probability that any of the classes are true. The softmax is given in equation 2.10.

$$f(net_j) = \frac{e^{net_j}}{\sum_{k=1}^{K} e^{net_k}} \tag{2.10}$$

Where K is the number of outputs, net_j is the preactivation value from neuron j, and K is all the outputs from that layer.

2.6.3 Feed forward neural networks

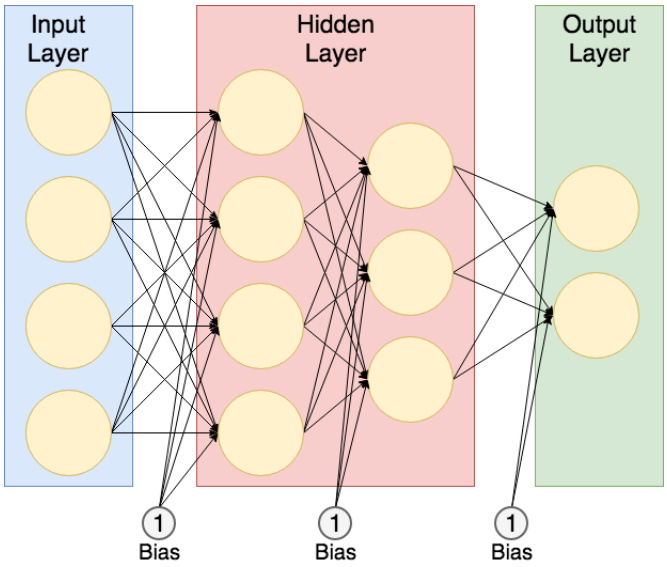


Figure 2.5: A fully connected feed forward network with two hidden layers.

In a feed-forward neural network, the data feeds only in one direction, from the input to the hidden layers to the output. All layers between the input and output are called hidden layers. Figure 2.5 shows a simple example of a neural network with two hidden layers, using only fully connected layers. In a fully connected layer all neurons in the layer are linked to all neurons in the previous layer, the net value from a single neuron is a function of all outputs in the previous layer.

2.6.4 Convolutional layers

There are circumstances where there is not useful for all the neurons to consider all the input from the previous layer, instead just consider a small part of the input. For example when using local patterns, like edges, corners, etc in an image to determine what kind of object there is in the image. A convolutional layer (CL) works like that and differ from a fully connected layer in that the neurons only receive input from a small neighbourhood of neurons in the previous layer. The neighbourhood is often referred to as the neurons *receptive field*, and its shape are given by the shape of the filter kernel, which often are much smaller than the total input to the layer. The weights of the filter kernel are also modifiable as for the weights in a fully connected layer. The CL often has multiple filters, where each filter captures different features of the input. The output from the filter kernels and activation function is often refereed to as feature or activation maps. This is especially useful when the input consist of local spatially patterns.

Another important trait of CL is weight sharing. While the weights in a fully connected layer is only used once and never revisited, the weights in the CL is used across

the input, thus dramatically increase the efficiency in terms of memory requirements and statistical efficiency [24].

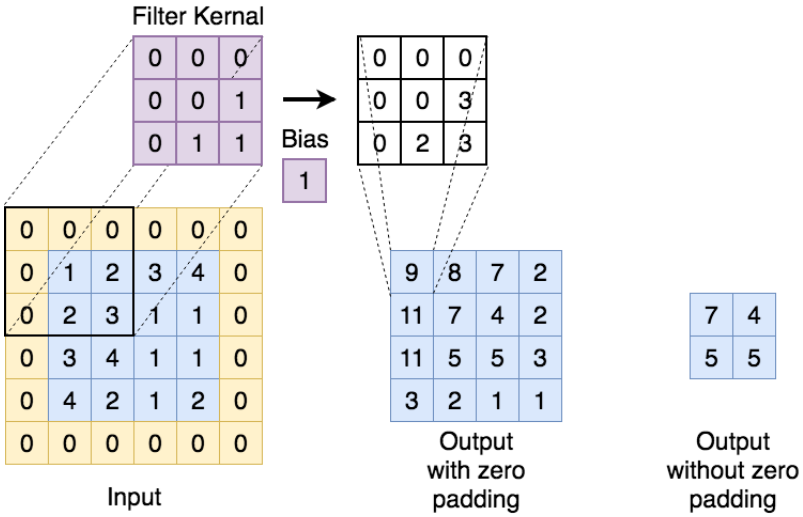


Figure 2.6: Example of a convolutional layer with 4x4 input, 3x3 filter kernel, one filter, stride of 1 in both dimensions, with and without 1x1 zero padding around the borders.

The hyperparameters for controlling CLs are filter kernel shape, number of filters, stride of each dimension and optional padding. Stride is how far the filter kernel moves between each convolution. Figure 2.6 shows an example of a CL with 4x4 input, 3x3 filter kernel, one filter, stride of 1 in both dimensions with and without zero padding.

The output size for each dimension from the CL are given from the input size and hyperparameters as:

$$\text{Output size} = \frac{\text{Input - filter + 2 x padding}}{\text{stide}} + 1 \tag{2.11}$$

In order to keep the input shape, the stride must be one and padding must be utilized. Networks which uses convolutional layers are often referred to as convolutional neural networks, CNN, or ConvNets. CNN often uses fully connected layers in the end to process or classify using the features captured in the feature maps.

2.6.5 Pooling layers

Pooling layers effectively reduces the dimension of the data, reducing the number of parameters and the risk of overfitting the data. Pooling layers are very common in CNN architectures. Examples of pooling layers are maxpool which only keeps the highest value in a region and avgpool which keeps the average of the region. Maxpooling is said to lead to faster convergence rate in vision recognition by selecting superior invariant features which improves generalization performance [25]. Figure 2.7 shows an example of pooling, using maxpool and avgpool.

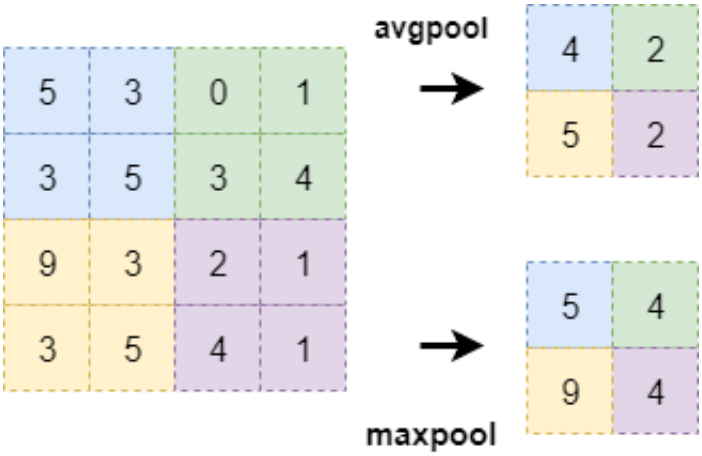


Figure 2.7: The result 2x2 avgpool and maxpool, with stride = 2 for both dimension, on a 4x4 input.

2.6.6 Training of neural networks

All the weights in neural network are often initialized to random values within some constraints, e.g. normal distributed with mean of zero and standard deviation of one, and the bias are often set to a positive small constant. The basis of the training or the learning of neural network, is that some training data pattern are presented to the input layer, runs through the network, and determine the output values. The output values are then compared to the target values, giving a cost value using some cost function. This cost is used to modify the weights in a direction that reduces the cost the most, thereby training the network to perform better. The magnitude of the modification to the weights depends on a given constant or adaptive learning rate.

The training is often done using mini-batch back-propagation, where training data is split into several mini-batches, with an equal batch-size. The weights of the network is updated after each mini-batch, by using back-propagation to estimate each weight's contribution to the cost, and a gradient decent optimizer to adjust the weight in a way that reduces the cost. The cost function is chosen depends on the task, e.g. classification or regression. For classification a common cost functions is cross-entropy.

2.6.7 Regularization

Neural networks are prone to overfitting, due to the sheer amount of free variables. Overfitting, also refereed to as overtraining, is when a model agrees very well with available data, but fails to perform when using unseen or new data. The best way to increase the generalization of a model is to increase the amount of training data, however this is not always possible and other techniques must be applied. There are several regularization techniques for neural networks to improve generalization and reduce the change of overfitting. Some common techniques are presented below.

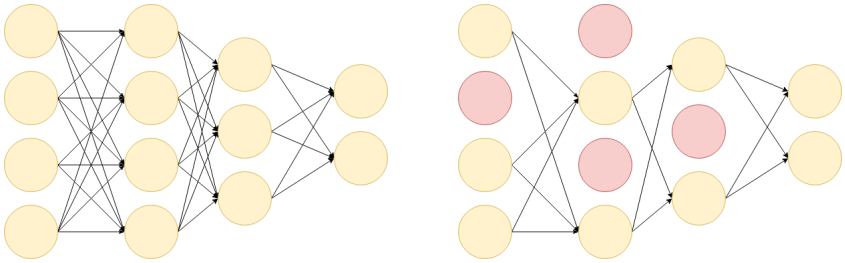


Figure 2.8: A simple example of dropout, where the connection to the red coloured neurons are dropped.

Early stopping

Early stopping is a form of regularization that require almost no change in the underlying training procedure, the cost function or the allowable set of parameter values. [24]. The idea is to stop training when the validation error is at the minimum, which hopefully will lead to better test results. In practice this can be achieved by saving the model each time the validation error is lower than the current lowest validation error. When the training terminates, the model with the lowest validation error is restored and tested with the test result. This is often combined with a terminate clause in the training, which terminates the training if the validation error is not improved in a pre defined number of iterations.

Dropout

Dropout makes sure that the model does not depends massively on one or a few features [26]. For each iteration (e.g. mini-batch), the neurons in a layer are kept with a keep probability p. For the neurons that are not kept, all the connections related to the neurons are dropped. During validation and test all the neurons are usually kept, and the outputs are scaled with p. Figure 2.8 illustrates dropout in a simple fully connected neural network.

2.7 Evaluation metrics

When the true label is available, e.g. in supervised learning, the confusion matrix is a common evaluation method. The confusion matrix collects the predicted and true labels, and organizes the results in a matrix with the true state labels in one axis, and the predicted labels in the other, as shown for a two class binary problem in Figure 2.9. The confusion matrix can be used to calculate overall accuracy, and precision and recall for each class.

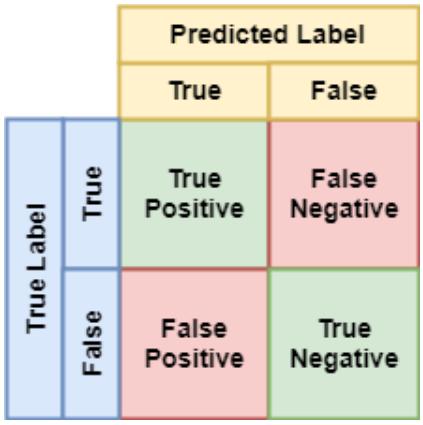


Figure 2.9: Binary confusion matrix.

The overall accuracy for a classification problem, can be expressed from the elements of a confusion matrix as:

$$\text{TotAcc} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}} \tag{2.12}$$

Where TP is the true positive, TN is true negative. FP is false positive, often refereed to as type I error. FN is false negative, often refereed to as type II error.

In the event of uneven sizes of the classes, or misclassification of a specific class is more costly, some other measurements can be useful for evaluation of the method. Precision (also known as postive predictive value) and recall (also known as sensitivity or true positive rate) for class k are given by equation 2.13 and 2.14 respectively.

$$\text{Precision}_k = \frac{\text{TP}_k}{\text{TP}_k + \text{FP}_k} \tag{2.13}$$

$$\text{Recall}_k = \frac{\text{TP}_k}{\text{TP}_k + \text{FN}_k} \tag{2.14}$$

The confusion matrix can easily be extended to a multi-class classification problem.

Chapter 3

Material and Methods

This chapter describes the data material used in this thesis and the methods used for pre-processing, feature extraction and classification of the MRI scans. The texture features are classified using RF, while for CNN there are no features extracted in advance.

3.1 Data Collection

ADNI

Data used in the preparation of this thesis were obtained from the Alzheimer's Disease Neuroimaging Initiative (ADNI) database. The ADNI begun in 2004 as a private-public partnership under the leadership of Dr. Michael W. Weiner. The primary goals of ADNI are[27]:

1. Detect AD at the earliest possible stage, and identify ways to track the disease using MRI, positron emission tomography (PET), other biological markers, and clinical and neuropsychological assessment, or a combination of those.

2. Support advances in AD prevention and treatment, through application of new diagnostic methods at the earliest possible stages.

3. Continually administer ADNI's data-access policy, which provides all data without embargo to all scientists in the world.

The initial ADNI study, ADNI-1, started in 2004 with a five year duration. The study included 200 elderly controls, 400 with MCI and 200 with AD. The study were later extended into ADNI-GO in 2009, ADNI-2 in 2011 and ADNI-3 in 2016[28]. The new studies included in addition to the existing subjects from ADNI-1, also new subjects. For up-to-date information see `www.adni-info.org`. In this thesis the subjects and data from ADNI-1 is used, but the meta data from the subjects are taken from all ADNI studies.

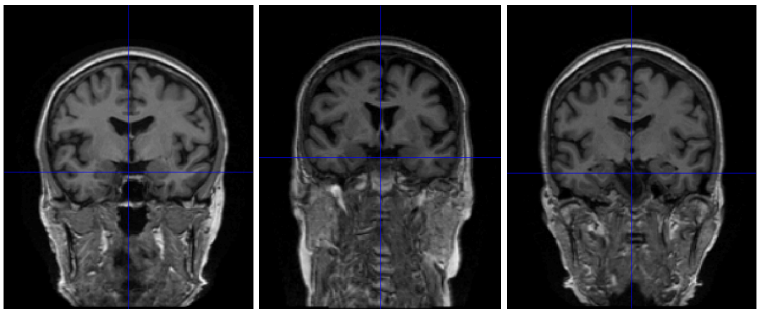
Table 3.1: ADNI 1 standardized data sets.

Data set	NC	MCI	AD	Visits
Screening	229	401	188	(Sc)
1 Year	195	311	133	(Sc, M06, M12)
Annual 2 Year	169	234	101	(Sc, M12, M24)
2 Year	168	212	99	(Sc, M06, M12, M18*, M24)
3 Year	135	148	99	(Sc, M06, M12, M18*, M24, M36**)

NC = Normal Control, MCI = Mild cognitive impairment, AD = Alzheimer's Disease, Sc = Screening, M06 (M12, M18, M24, M36) = Month 6 (12, 18, 24, 36), *M18 - only for MCI, **M36 - only for NC and MCI

Data material

The data material used in this thesis consist of T1- weighted MRI with a field strength of 1.5T, and is collected from the standardized 1.5T data sets from ADNI 1 [29]. The standardized data sets includes 818 subjects at screening with up to three years follow up visits, diagnosed with either MCI or AD, in addition to NC. The visits are at the intervals: screening, Month 6, Month 12, Month 18[1], Month 24 and Month 36[2]. Table 3.1 shows the number of subjects for each diagnostic group included in each of the five standardized sets at screening. Some examples of the MRI scans from the data material is visualized in Figure 3.1, which shows one example subject from the three diagnostic groups NC, MCI and AD. In addition to the MRI data, meta data for each subject are also available from ADNI, e.g. diagnosis at each visit, age, education years, mini-mental state examination (see Table 3.2).



(a) NC (b) MCI (c) AD

Figure 3.1: Example of the MRI scans from the different diagnostic groups in the data material, from the coronal plane.

Of the 818 subjects at screening, one AD subject was missing from the set when the

[1]Only MCI subjects.
[2]Only NC and AD subjects.

data was downloaded[3], and therefore not included in this data set, and 5 MCI subjects were also excluded because there where a mismatch between the baseline diagnosis and the diagnosis at the first screening visit. This results in a total of 811 subjects which are grouped in three diagnostic groups at screening, AD, MCI and NC.

For each subject the MRI scans from all visits up to 36 months are included, if no diagnosis change from the screening diagnosis are registered. In case of a diagnosis change, only the scans for the visits until the last non diagnosis change visit are included. Thus for each subject there are 1-6 MRI scans in the data set. This result in a total of 2688 MRI scans from 811 subjects, see Table 3.2.

Diagnostics groups

In order to investigate how well early detection of AD can be identified, the MCI subjects are divided into three groups, MCI stable (MCIs), MCI convert (MCIc) and MCI other (MCIo). For MCIs the diagnosis must be MCI at screening and the diagnosis for the subject must not have changed in any of the following visits recorded by ADNI, which for some subjects includes up to 144 months follow up visits[4], and they must have conducted at least a 24 months follow up visit. The MCIc group consists of subjects who have converted from MCI to AD by 36 months and not reverted back to MCI or NC during any later follow up. The MCIo group are the MCI subjects which does not fit in the other two categories, because either they don't have a 24 months follow up, they reverted to NC or they converted to AD and reverted back later. The NC and AD groups are the same as diagnosed at screening. The number of subjects in each of the 5 diagnostic groups is shown in Table 3.2, the MCI group includes all of the MCIs, MCIc and MCIo groups.

Table 3.2: Overview of the included subjects at screening with total number of scans in brackets, age are reported as mean with standard deviation in brackets.

Diagnosis	Subjects (Scans)	Age	Sex (M/ F)
NC	228 (895)	75.9 (5.0)	118/ 110
AD	187 (545)	75.2 (7.5)	98/ 89
MCI	396 (1248)	74.7 (7.4)	255/ 141
MCIs	99 (437)	75.1 (7.0)	62/ 37
MCIc	166 (437)	74.7 (6.8)	100/ 66
MCIo	131 (374)	74.5 (8.3)	93/ 38

[3]Downloading date: 07.02.18
[4]At the time of meta files download: 07.02.18

3.2 Methods

Figure 3.2 shows a simple overview of the procedure used for classification of MRI in this thesis. All MRI images are pre-processed using different methods, and classified either using different feature extraction methods and RF, or with CNN.

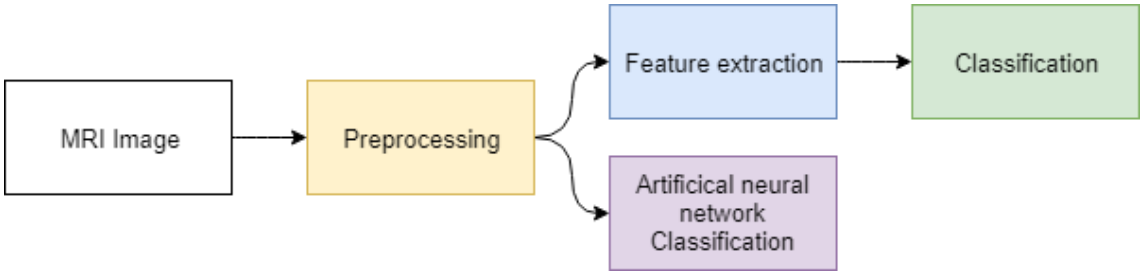


Figure 3.2: Flow diagram of the methods used.

3.2.1 Pre-processing

The image data downloaded from ADNI has already undergone some pre-processing steps. This was done in order to limit the differences on the MRI from the different scanning sites, and to limit the number of pre-processing strategies in the research literature, which leads to better comparability between different research methods. The pre-processing steps from ADNI includes[30]:

1. **Gradwarp** - Gradwarp is correction of image geometry distortion due to gradient non-linearity. This correction is system specific.

2. **B1 non-uniformity** - This correction procedure uses B1 calibration scans to correct the image non-uniformity that happens when the RF transmission is performed with a more uniform body coil while reception is performed with a less uniform head coil.

3. **N3** - N3 is a histogram peak sharpening algorithm. It is applied after gradwarp and the B1 correction above.

The further pre-processing steps in this thesis are done using the SPM12[5] toolbox in MATLAB. Figure 3.3 gives an overview over the pre-processing steps. Briefly the pre-processing consists of: normalization of the MRI, segmentation, skull-stripping and gaussian smoothing.

[5]Available from: `http://www.fil.ion.ucl.ac.uk/spm/software/spm12/`

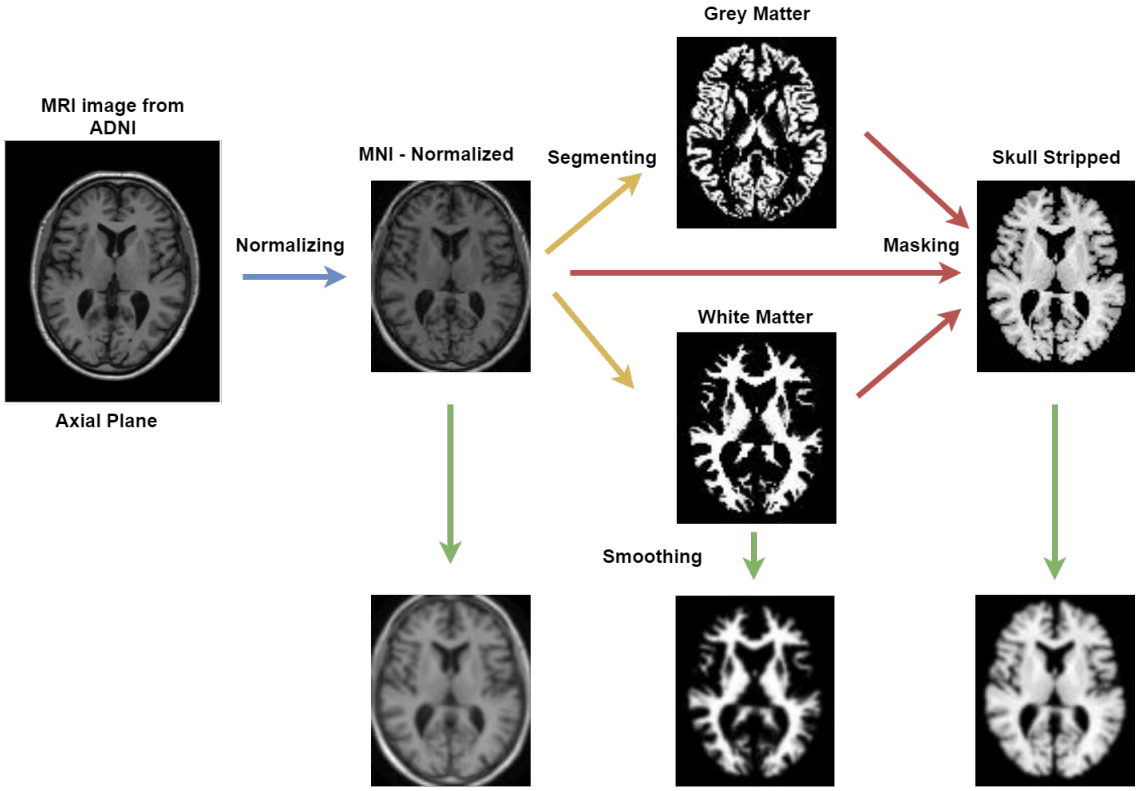
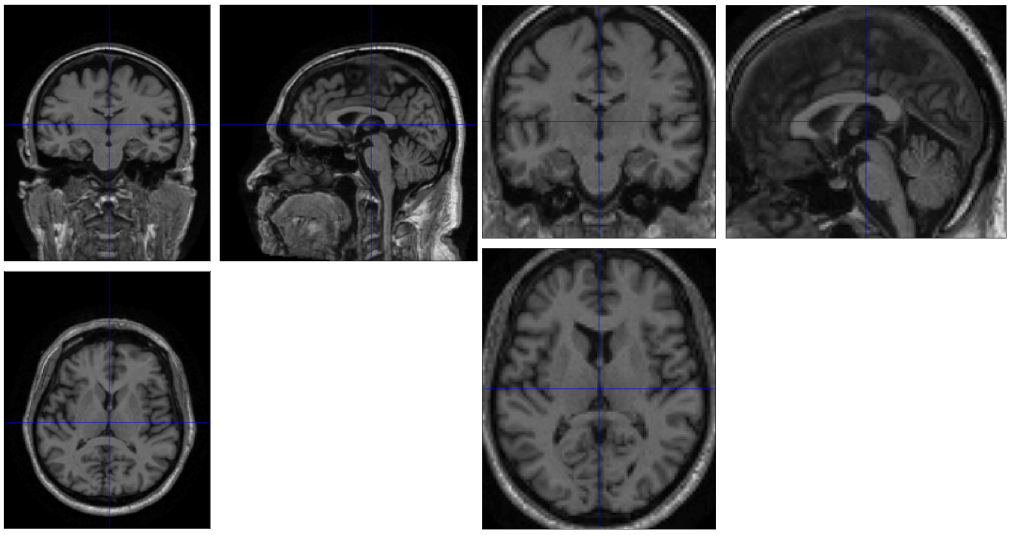


Figure 3.3: Overview of the pre-processing methods.

Normalization

Before normalization the images had varying resolution and voxel size, the resolution varied from 192x192x160 to 256x256x166. The MRI scans were normalized to MNI152 (Montreal Neurological Institute) space[31]. The MNI152 brain template was created from the 3D brain MRI of 152 normal subjects [32]. The normalization is done using the normalization procedure in SPM12, with default settings. After normalization all images had the resolution 79x95x79 with a voxel size of 2x2x2mm. Figure 3.4 shows an example of a subject MRI before and after normalization.



(a) (b)

Figure 3.4: Example of the normalization to MNI space for a subject shown in the axial plane. (a) Before normalization, (b) After normalization to MNI space.

Segmentation

After the normalization the normalized MRI images are segmented into grey matter (GM), WM, CSF, bone, soft tissue and background, using the standard segmentation procedure in SPM12.

Skull stripping

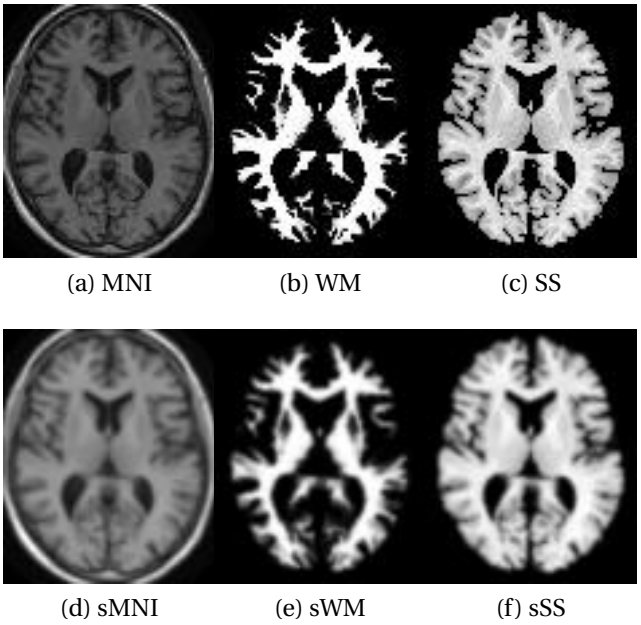
For skull stripping, the sum of the segmented GM and WM, with a threshold value, are used as a mask to skull strip the MRI.

$$I_{SS} = I_{MNI} .* ((I_{GM} + I_{WM}) > T) \tag{3.1}$$

I_{SS} is the skull stripped image, I_{MNI} is the MNI152 normalized brain image and I_{WM} and I_{GM} are the segmented brain images, T is the threshold value. Higher threshold value gives an more stringent skull stripping. In this thesis the threshold value is set to 0.2. An example of the result of the skull stripping can be seen in top rightmost image in Figure 3.3.

Smoothing

The smoothing is done using a full width at half maximum Gaussian smoothing kernel using a kernel size of $3x3x3$ mm. Figure 3.5 illustrates the result from this prepossessing step, and the examples of the 6 pre-processing methods that was tested in this thesis.



(a) MNI (b) WM (c) SS

(d) sMNI (e) sWM (f) sSS

Figure 3.5: Overview of the pre-processing methods used in this thesis.

3.2.2 Feature extraction

LBP-TOP features

The LBP-TOP features was extracted using MATLAB with LBP-TOP[6] developed by Zhao and Pietikäinen [19]. Bilinear interpolation was used with the radius, number of neighboring points and both rotation invariant (ri) and uniform rotation invariant (riu2) LBP patterns as specified in Table 3.3. The neighboring points and radius was the same for all dimensions. The features was extracted by concatenating the normalized histograms of the whole LBP image from the coronal, axial and sagittal plane.

Table 3.3: Overview of the different parameters used in LBP-TOP features extraction.

R	P	Pattern	Num features
1	8	riu2/ ri	30/ 108
2	12	riu2/ ri	42/ 1056
3	16	riu2/ ri	54/ 12348
all		riu2/ ri	126/ 13512

R is radius, P is number of neighboring points and pattern is either rotation invariant and uniform (riu2) or rotation invariant (ri), all is the concatenated histogram of (r=1 & p=8), (r=2 & p=12) and (r=3 & p=16)

In addition to the LBP-TOP histograms of the entire image, there were also tested to

[6]Available from: http://www.cse.oulu.fi/CMV/Downloads/LBPMatlab

split the LBP-TOP images into cells of equal size, LBP-TOP cells. LBP-TOP histograms from each cell is concatenated into a final histogram, illustrated in Figure 3.6. This was only tested with the uniform rotation invariant pattern, to limit the amount of features. For the combination of three sets with radius and neighboring points, the histograms are also concatenated cell wise.

Table 3.4: Overview of the different parameters used in LBP-TOP cells features extraction.

c	R	P	Pattern	Num features
15/ 18	1	8	riu2	4500/ 2400
15/ 18	2	12	riu2	6300/ 3360
15/ 18	3	16	riu2	8100/ 4320
15/ 18	all		riu2	18900/ 10080

c is cell size in voxels. R is radius, P is number of neighboring points, all is the concatenated histogram of (r=1 & p=8), (r=2 & p=12) and (r=3 & p=16)

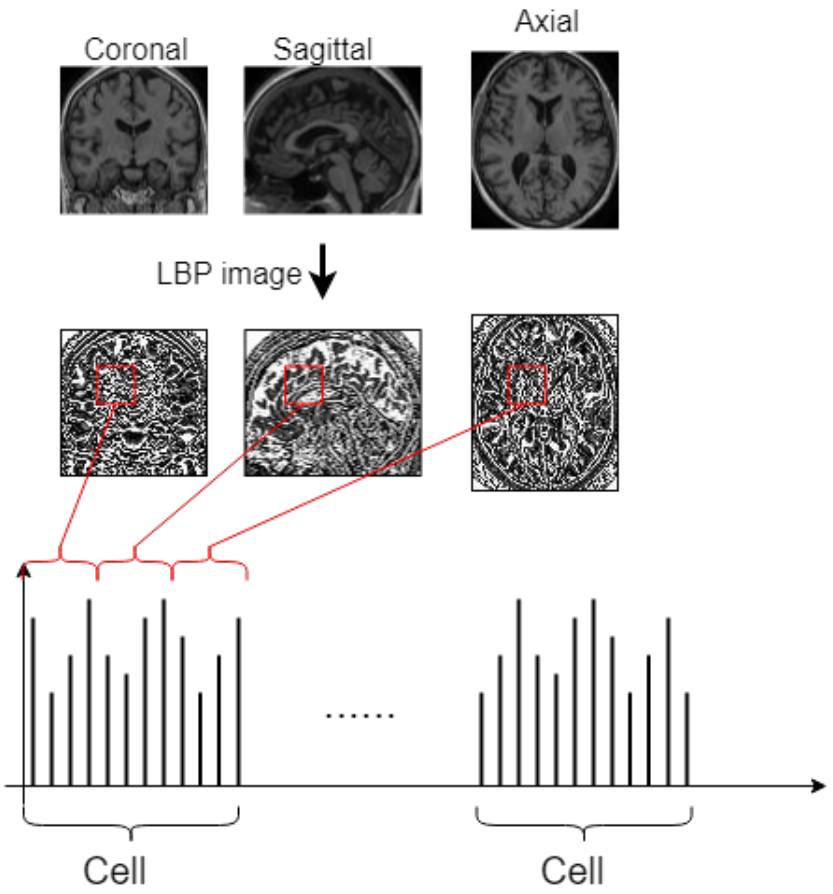


Figure 3.6: Illustration of the LBP-TOP cells.

3D VHOG features

The 3D VHOG features are extracted using MATLAB implementation "Histogram of Orientated Gradients 3D (3D Voxel HOG)[7] by R. Dupre, which was originally developed for local object structure detection, for use in risk analysis framework [21, 33]. The parameters used in this thesis for extracting the features are summarized in Table 3.5.

Table 3.5: The parameters used for 3D VHOG feature extraction.

c	b	s	Num features
15	1	1	6750
15	2	1	28800
18	1	1	3600
18	2	1	12960

c is cell size given in voxels, b is block size given in cells, s is stride in cells

Combination of LBP-TOP and 3D HOG features

A combination of 2D HOG and LBP have improved the performance of some classifiers, compared to HOG and LBP alone, e.g. in human detection [34]. Therefore the combination of LBP-TOP and 3D VHOG is tested to see if it will improve the accuracy in this MRI data set. The combination of LBP-TOP and 3D VHOG features, are made by cell wise concatenate the LBP-TOP histogram and 3D VHOG histogram. The 3D VHOG features extracted using a cell size of 15 and 18, and a block size of 1, and is concatenated with LBP features using r=1 & p=8, r=2 & p=12, r=3 & p=16 and and a concatenated histogram of these, using the same cell size as for 3d VHOG. This is summarized in Table 3.6.

Table 3.6: Overview of the different parameters used in the combination of LBP-TOP and 3D VHOG features.

c	R	P	Pattern	Num features
15/ 18	1	8	riu2	11250/ 6000
15/ 18	2	12	riu2	13050/ 6960
15/ 18	3	16	riu2	14850/ 7920
15/ 18	all		riu2	25650/ 13680

c is cell, size, R is radius, P is number of neighboring points and pattern is either rotation invariant and uniform (riu2) or rotation invariant (ri), all is the concatenated histogram of (r=1 & p=8), (r=2 & p=12) and (r=3 & p=16)

[7] Available from: https://www.mathworks.com/matlabcentral/fileexchange/55978-histogram-of-orientated-gradients-3d--3d-voxel-hog-

Table 3.7: Overview over number of subjects and scans of each class in the classification problems tested.

classification problem	Tr0 Te0	Tr1 Te1	Tr2 Te2	Tr3 Te3	Sum
NC/ MCIs/ MCIc/ AD	190 (385) 38 (48)	89 (385) 10 (48)	148 (385) 18 (48)	151 (385) 36 (48)	578 (1540) 102 (192)
NC/ MCI/ AD	194 (472) 34 (73)	332 (472) 64 (73)	163 (472) 24 (73)	-	689 (1416) 122 (219)
NC/ AD	190 (468) 38 (77)	162 (468) 25 (77)	-	-	352 (936) 63 (154)
NC/ MCI	194(767) 34 (128)	336 (767) 60 (128)	-	-	530 (1534) 94 (256)
MCI/ AD	340 (450) 56 (95)	155 (450) 32 (95)	-	-	495 (900) 88 (190)
MCIs/ MCIc	83 (369) 16 (59)	142 (369) 24 (59)	-	-	225 (738) 40 (118)

TrX, TeX are training and test data respectively for class number X, and are given with the number of subjects and number of scans in brackets. The class number are in the order they are listed under classification problem.

3.2.3 Classification

The classification is split up into different classification problems, 4-, 3-class and binary problems. The four-class is: NC vs MCIs vs MCIc vs AD, three-class: NC vs MCI vs AD, and the binary problems: NC vs AD, NC vs MCI, MCI vs AD and MCIs vs MCIc. The four-class problem and the binary problem, MCIs vs MCIc, is the main focus in this thesis as they are a measure on how well early detection of AD can be identified.

For each classification problem the subjects are split into a training set, 85%, and a test set, 15%. These sets are the same for all methods tested. The test set must not be confused with the validation set. The test set in this thesis are the used to evaluate how well the methods generalizability to unseen data. The validation set is taken from the training data in each method, to choose the best performing approach for the task for each method, e.g. which parameters/ hyperparamers best suited for the task.

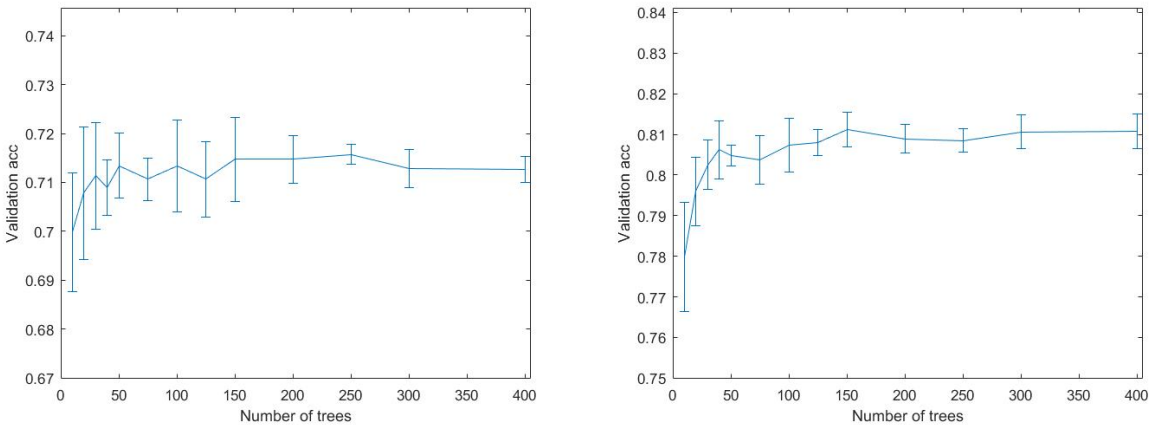
Additional visit scans are added in each set (original training and test set) until the number of scans in each class is equal to the minimum number of available additional scans for a class, with priority to the visits closest to the screening. This assures equal number of instances of each class in the training and test set, and that no visit scan from the same subject are in both the training and test set, which is done to minimize the data leakage. Table 3.7 shows the number of subjects and scans in both the training and test set for each classification problem.

A RF classifier is used for classification of the features extracted from the pre-processed

MRI. For neural network there is no feature extraction done before classification, bur the same pre-processing methods are tested. The different pre-processing methods for the MRI tested are the MNI, smoothed MNI, WM, smoothed WM, skull stripped and smoothed skull stripped, which are illustrated in Figure 3.5.

Random Forest

The features extracted using the methods presented in Section 3.2.2 are classified using a RF classifier, using the implementation from the scikit learn library package in python3.6.[8]. The parameters for the RF is set at default values, except for the number of trees in the classifier.For the maximum amount of trees to be tested, an initial test using relatively few features and many features was conducted on the NC vs AD problem. Figure 3.7 shows the mean of the mean 10-fold cross validation result using up to 400 trees, with the standard deviation as error bars, using the 126 features from LBP-TOP (a) and the 25650 features from the combination of LBP-TOP and 3D VHOG (b). Since there seemed to be a small gain up to 150 trees, all test were conducted with 10, 30, 50, 75, 100 and 150 trees. No feature selection or feature reduction is performed before the RF classifier.



(a) 126 features from LBP-TOP

(b) 25650 features from LBP-TOP + 3D VHOG

Figure 3.7: Effect of number of trees in random forest on validation results

Neural Network

The neural network tested in this thesis is implemented using python3.6 and the library package Tensorflow[9]. All networks were trained by splitting the training set into a new training set (85% of the original training set) and a validation set (15% of the original training set), with a single subjects scans only being in either the training or validation

[8]Available from: `http://scikit-learn.org/stable/index.html`

[9]Available from: `https://www.tensorflow.org/install/`

set. The test set used for neural networks is equal to the test set that were used with the RF classification of features. Because of the excessive computational cost related to 3D CNN, there is a limited set of hyper parameters tested. The networks tested differ by different number of CL, and different dropout in the last fully connected layer. The models tested were:

- **6 3D convolutional layers**: The first two CL have 8 filters and a kernel size of $6x6x6$, the next two CL have 16 filters and a kernel size of $5x5x5$, the last two CL have 32 filters and a kernel size of $3x3x3$. There is also a $2x2x2$ maxpooling layer after the 2nd CL, and a $3x3x3$ maxpooling layer after the 5th CL.
- **5 3D convolutional layers**: The first two CL have 8 filters and a kernel size of $6x6x6$, the next two CL have 16 filters and a kernel size of $5x5x5$, the last CL have 32 filters and a kernel size of $3x3x3$. There is also a $2x2x2$ maxpooling layer after the 2nd CL, and a $3x3x3$ maxpooling layer after the 4th CL.
- **4 3D convolutional layers**: The first CL have 8 filters and a kernel size of $6x6x6$, the next two CL have 16 filters and a kernel size of $5x5x5$, the last CL have 32 filters and a kernel size of $3x3x3$. There is also a $2x2x2$ maxpooling layer after the 1st CL, and a $3x3x3$ maxpooling layer after the 3rd CL.
- **3 3D convolutional layers**: The first CL have 8 filters and a kernel size of $6x6x6$, the next CL have 16 filters and a kernel size of $5x5x5$, the last CL have 32 filters and a kernel size of $3x3x3$. There is also a $2x2x2$ maxpooling layer after the 1st CL, and a $3x3x3$ maxpooling layer after the 2nd CL.

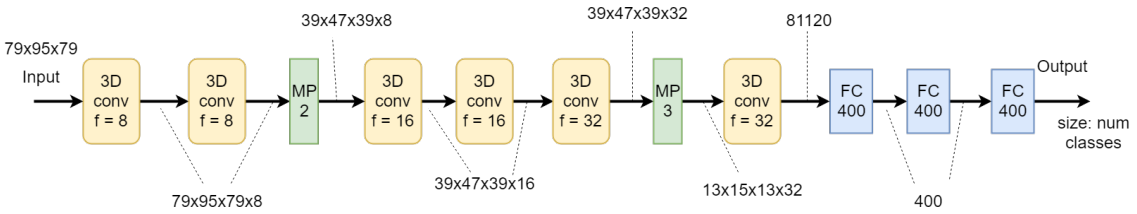


Figure 3.8: Convolutional neural network with 6 convolutional layers and 3 fully connected layers.

For all CL there were used zero-padding, stride of one, and ReLU as the activation function. All networks were tested with 3 fully connect layers after the convolution layer, with 400 neurons in each and ReLU as the activation function. The learning was set to 10^{-4} since this gave the best result in initial testing. Batch size was set to 16 due to limitation of GPU memory, smaller batch sizes were also initially tested but this lead reduced training speed without increasing the classification performance of the network. All networks were tested with 0, 10%, or 20% dropout in the last fully connected layer. Figure 3.8 shows an illustration of the network with 6 convolutional layers.

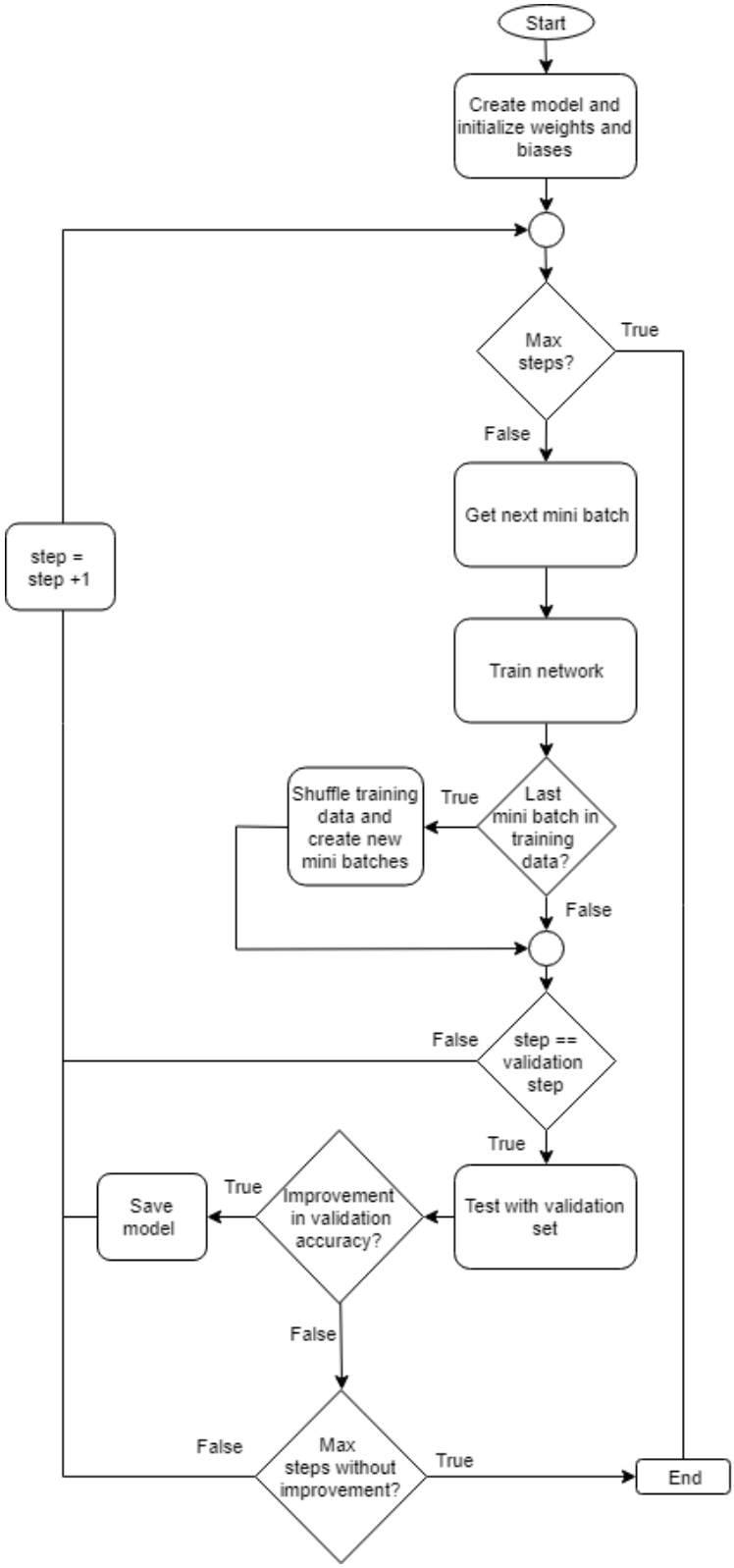


Figure 3.9: Illustration of the training of the neural networks

For training of the networks, all weights are initialized with a truncated normal distribution and the biases are initialized to 0.1. The training procedure of the CNN are

illustrated in Figure 3.9. For each iteration or step, a new mini batch is used to train the network. The networks performance is tested with the validation set at fixed steps, which is set every $validation\_step$ given by equation 3.2.

$$validation\_step = floor(\frac{training\_data}{batch\_size}) \tag{3.2}$$

If the accuracy on the validation data is better than the last best validation accuracy, the network is saved. When the all the mini batches in the training data has been used or the last mini batch is smaller than the specified batch size, the training data is reshuffled and split into new mini batches. The training continues until a predefined max steps without improvement or max steps are reached. When the network terminates training, the network parameters with the best validation accuracy are loaded and tested with the test set.

Chapter 4

Results

In this chapter the results from the methods described in chapter 3 are presented.

Section 4.1 presents the best validation results from each pre-processing method and all classification problems for each feature extraction method tested and for CNN. For each feature extraction method and CNN for all classification problems, the pre-processing method with the best validation accuracy is highlighted.

The highlighted methods (best performing methods) are presented with the results from the test set in section 4.2.

4.1 Validation results

Table 4.1 presenters the the best validation accuracy for each pre-processing, classification problem and method tested. More detailed tables of the validation results can be found in appendix A, where also the hyperparameters used is presented.

From table 4.1 it can be seen that the skull stripping pre-processing achieved the best results for all the methods tested. Although there were not a huge differences between the validation accuracy between the smoothed skull stripped and skull stripped images, the non-smoothed skull stripped images performed better for almost all feature extraction methods.

Table 4.1: The best validation result for each pre-processing, classification problem and classification method tested. The validation result for LBP, HOG and LBP+HOG are given as the mean of 10-fold cross validation. The best validation accuracy for each classification method and classification problem are in bold.

Test	4-class	3-class	NC/ AD	NC/ MCI	MCI/ AD	MCIs/ MCIc
LBP: SS	**0.403**	**0.552**	**0.819**	**0.611**	**0.673**	**0.566**
LBP: sSS	0.377	0.544	0.802	0.600	0.620	0.561
LBP: WM	0.332	0.469	0.707	0.600	0.620	0.542
LBP: sWM	0.356	0.505	0.754	0.563	0.636	0.536
LBP: MNI	0.352	0.490	0.725	0.555	0.644	0.535
LBP: sMNI	0.349	0.498	0.736	0.562	0.628	0.535
HOG: SS	0.390	**0.548**	**0.811**	**0.609**	**0.671**	0.577
HOG: sSS	**0.392**	0.527	0.798	0.600	0.646	**0.597**
HOG: WM	0.341	0.495	0.729	0.583	0.641	0.543
HOG: sWM	0.360	0.501	0.767	0.601	0.611	0.543
HOG: MNI	0.362	0.511	0.766	0.582	0.622	0.548
HOG: sMNI	0.353	0.481	0.774	0.611	0.611	0.530
LBP+HOG: SS	**0.396**	**0.555**	**0.815**	**0.612**	**0.670**	**0.576**
LBP+HOG: sSS	0.385	0.542	0.804	0.601	0.666	0.574
LBP+HOG: WM	0.338	0.484	0.714	0.597	0.636	0.534
LBP+HOG: sWM	0.358	0.495	0.746	0.588	0.636	0.530
LBP+HOG: MNI	0.346	0.505	0.735	0.579	0.639	0.521
LBP+HOG: sMNI	0.363	0.503	0.735	0.577	0.624	0.544
CNN: SS	0.375	0.592	**0.891**	0.656	**0.786**	0.667
CNN: sSS	**0.448**	**0.615**	0.836	**0.729**	0.750	**0.787**
CNN: WM	0.388	0.567	0.781	0.633	0.740	0.675
CNN: sWM	0.394	0.567	0.820	0.642	0.698	0.725
CNN: MNI	0.365	0.558	0.781	0.708	0.708	0.708
CNN: sMNI	0.427	0.500	0.789	0.625	0.729	0.625

SS = skull stripped image, WM = white matter image, MNI = MNI normalized image,s = smoothed image LBP is LBP-TOP cells, HOG is 3D VHOG, LBP+HOG is LBP+TOP and 3D VHOG combined

4.2 Test results for the classification problems

The methods with the best validation accuracy are presented with the classification results from the unseen test set. The unseen test set is not used for training or to choose which hyper parameters to be used, and is a measurement how well the methods generalizes. The best method are in bold for each classification problem.

In all tables below, LBP is LBP-TOP cells, HOG is 3D VHOG, and LBP+HOG is the combination between LBP-TOP cells and 3D VHOG.

4.2.1 NC vs MCIs vs MCIc vs AD

Table 4.2 shows the results using the test set for the NC vs MCIs vs MCIc vs AD classification problem. The accuracy from the test set are very similar to the validation accuracy for all the feature extraction methods with RF, but with CNN there is a clear difference, which can indicate overfitting for CNN. From the table it can also be seen that the most of the classifiers struggles to classify MCIs and MCIc correctly. The best test result was achieved with LBP-TOP cells, at 41.1%.

Table 4.2: Test results from the 4-class classification problem, NC vs MCIs vs MCIc vs AD.

Test	AccVal	AccTest	P0	P1	P2	P3
			R0	R1	R2	R3
$LBP^{SS,ntr:150}_{c15,R3}$	0.403(0.059)	**0.411**	0.369	0.000	0.484	0.575
			0.854	0.000	0.312	0.479
$HOG^{sSS,ntr:100}_{c15,b2}$	0.392(0.047)	0.391	0.356	0.250	0.400	0.462
			0.750	0.021	0.292	0.500
$LBP+HOG^{SS,ntr:150}_{c18,R1}$	0.396(0.048)	0.380	0.358	0.000	0.190	0.812
			0.812	0.000	0.083	0.625
$CNN^{sSS}_{CL6,d0.1}$	0.448	0.354	0.355	0.426	0.333	0.250
			0.229	0.542	0.500	0.149

AccVal is validation accuracy and given in mean with standard deviation in brackets. AccTest is the accuracy for the test set, Px, Rx are precision and recall for class x. 0 is NC, 1 is MCIs, 2 is MCIc, 3 is AD, c is cell size, b is block size in cells, R1 = (R=1 and P=8), R3 = (R=3 and P=16), ntr = number of trees in random forest, CL is number of convolution layers, d is dropout, SS = skull stripped image and sSS = smoothed skull stripped image

4.2.2 MCIs vs MCIc

Table 4.3 shows the results using the test set for the MCIs vs MCIc classification problem. Even though the best test accuracy, 60.3%, was achieved with a CNN, there is a clear difference between the validation and test accuracy which can indicate that the CNN is overfitted.

Table 4.3: Test results from the binary classification problem, MCIs vs MCIc.

Test	AccVal	AccTest	P0 R0	P1 R1
$LBP^{SS,ntr:100}_{c18,R1}$	0.566(0.110)	0.602	0.667 0.407	0.407 0.797
$HOG^{sSS,ntr:50}_{c15,b2}$	0.597(0.118)	0.576	0.645 0.339	0.552 0.814
$LBP + HOG^{SS,ntr:50}_{c18,all}$	0.576(0.102)	0.559	0.652 0.254	0.537 0.864
$CNN^{sSS}_{CL4,d0.1}$	0.787	**0.603**	0.613 0.559	0.594 0.647

AccVal is validation accuracy and given in mean with standard deviation in brackets. AccTest is the accuracy for the test set, Px, Rx are precision and recall for class x. 0 is MCIs, 1 is MCIc, c is cell size, b is block size in cells, R1 = (R=1 and P=8), all = (R=1 and P=8) and (R=2 and P=12) and (R=3 and P=16) combined, ntr = number of trees in random forest, CL is number of convolution layers, d is dropout, SS = skull stripped image and sSS = smoothed skull stripped image

4.2.3 NC vs MCI vs AD

Table 4.4 shows the results using the test set for the NC vs MCI vs AD classification problem. The best test result, 48.9% was achieved with the 3D VHOG feature extraction, however there is a notable difference between the test and validation accuracy for all methods, especially for CNN.

Table 4.4: Test results from the 3-class classification problem, NC vs MCI vs AD.

Test	AccVal	AccTest	P0	P1	P2
			R0	R1	R2
$LBP^{SS,ntr:150}_{c15,all}$	0.552(0.052)	0.457	0.549	0.321	0.518
			0.616	0.356	0.397
$HOG^{SS,ntr:100}_{c15,b2}$	0.548(0.056)	**0.489**	0.512	0.369	0.569
			0.575	0.329	0.562
$LBP + HOG^{SS,ntr:150}_{c15,all}$	0.555(0.058)	0.461	0.511	0.329	0.541
			0.616	0.315	0.452
$CNN^{sSS}_{CL6,d0.1}$	0.615	0.470	0.559	0.382	0.560
			0.452	0.575	0.384

AccVal is validation accuracy and given in mean with standard deviation in brackets. AccTest is the accuracy for the test set, Px, Rx are precision and recall for class x. 0 is NC, 1 is MCI, 2 is AD, c is cell size, b is block size in cells, all = (R=1 and P=8) and (R=2 and P=12) and (R=3 and P=16) combined, ntr = number of trees in random forest, CL is number of convolution layers, d is dropout, SS = skull stripped image and sSS = smoothed skull stripped image

4.2.4 NC vs AD

For NC vs AD the best method, a combination of LBP-TOP cells and 3D VHOG, achieved a test accuracy of 82.5% as shown in Table 4.5. Again there is a notable difference between the test and validation accuracy for CNN.

Table 4.5: Test results from the binary classification problem, NC vs AD.

Test	AccVal	AccTest	P0 R0	P1 R1
$LBP^{SS,ntr:75}_{c15,all}$	0.819(0.059)	0.805	0.805 0.805	0.805 0.805
$HOG^{SS,ntr:75}_{c18,b2}$	0.811(0.063)	0.779	0.759 0.803	0.818 0.740
$LBP+HOG^{SS,ntr:50}_{c15,R1}$	0.815(0.060)	**0.825**	0.812 0.844	0.838 0.805
$CNN^{SS}_{CL6,d0.2}$	0.891	0.812	0.808 0.818	0.816 0.805

AccVal is validation accuracy and given in mean with standard deviation in brackets. AccTest is the accuracy for the test set, Px, Rx are precision and recall for class x. 0 is NC, 1 is AD, c is cell size, b is block size in cells, R1 = (R=1 and P=8), all = (R=1 and P=8) and (R=2 and P=12) and (R=3 and P=16) combined, ntr = number of trees in random forest, CL is number of convolution layers, d is dropout, SS = skull stripped image and sSS = smoothed skull stripped image

4.2.5 NC vs MCI

The test results from NC vs MCI are shown in Table 4.6. The test accuracy differences between feature extraction methods are small, between 64.1 - 67.6%, with the exception of CNN, which had the best validation result and lowest test result. This can indicate overfitting.

Table 4.6: Test results from the binary classification problem, NC vs MCI.

Test	AccVal	AccTest	P0 R0	P1 R1
$LBP^{SS,ntr:75}_{c15,all}$	0.611(0.063)	0.648	0.686 0.747	0.711 0.621
$HOG^{SS,ntr:150}_{c15,b2}$	0.609(0.042)	**0.676**	0.723 0.570	0.645 0.781
$LBP + HOG^{SS,ntr:150}_{c15,R2}$	0.612(0.058)	0.641	0.676 0.539	0.617 0.742
$CNN^{sSS}_{CL6,d0.2}$	0.729	0.559	0.574 0.459	0.549 0.659

AccVal is validation accuracy and given in mean with standard deviation in brackets. AccTest is the accuracy for the test set, Px, Rx are precision and recall for class x. 0 is NC, 1 is MCI, c is cell size, b is block size in cells, R2 = (R=2 and P=12), all = (R=1 and P=8) and (R=2 and P=12) and (R=3 and P=16) combined, ntr = number of trees in random forest, CL is number of convolution layers, d is dropout, SS = skull stripped image and sSS = smoothed skull stripped image

4.2.6 MCI vs AD

The test results from NC vs MCI, shown in Table 4.6, are quite similar to the NC vs MCI problem. The test accuracy differences between feature extraction methods are small, between 63.7 - 67.9%, with the exception of CNN, which had the best validation result and lowest test result. This can again indicate overfitting.

Table 4.7: Test results from the binary classification problem, MCI vs AD.

Test	AccVal	AccTest	P0 R0	P1 R1
$LBP^{SS,ntr:150}_{c18,R1}$	0.673(0.048)	0.653	0.641 0.695	0.667 0.611
$HOG^{SS,ntr:75}_{c15,b2}$	0.671(0.064)	**0.679**	0.663 0.726	0.698 0.632
$LBP + HOG^{SS,ntr:100}_{c15,R1}$	0.670(0.064)	0.637	0.623 0.695	0.633 0.688
$CNN^{SS}_{CL6,d0.2}$	0.786	0.580	0.565 0.696	0.605 0.464

AccVal is validation accuracy and given in mean with standard deviation in brackets. AccTest is the accuracy for the test set, Px, Rx are precision and recall for class x. 0 is MCI, 1 is AD, c is cell size, b is block size in cells, R1 = (R=1 and P=8), ntr = number of trees in random forest, CL is number of convolution layers, d is dropout, SS = skull stripped image and sSS = smoothed skull stripped image

Chapter 5

Discussion and Conclusion

In this chapter the results presented in Chapter 4 will be discussed, and a conclusion of this thesis and recommendations for future work is given.

Even though direct comparison between studies can be difficult, since different data sets often are used, the overall performance of the methods tested in this thesis is in the lower end of the comparable research studies. The best test accuracy achieved was 41.1% for NC vs MCIs vs MCIc vs AD, 60.3% for MCIs vs MCIc, 48.9% for NC vs MCI vs AD, 82.5% for NC vs AD, 67.6% for NC vs MCI and 67.9% for MCI vs AD.

5.1 Early detection of AD

Early detection of AD proved to be to complicated for the methods tested in this thesis. For the four-class differential diagnosis classification problem, with NC, MCIs, MCIc and AD, the best achieved test accuracy was 41.1% (Table 4.2) using LBP-TOP cells features, and it had problems in discriminating between MCIs and MCIc.

For the MCIs vs MCIc problem, the best method, CNN achieved a test accuracy of 60.3%, however this had a much higher validation accuracy, which indicates poor generalizability. All the feature extraction methods obtained quite similar validation and test accuracy to each other, and all of them had a quite high standard deviation on the validation accuracy from the 10-fold cross validation.

To the author's knowledge no paper has been published that reliable and automatically discriminates between the four groups NC, MCIs, MCIc and AD. And that it still is much work to be done in optimizing this multi-class problem.

5.2 Texture features vs CNN

There was no clear winner between the different feature extraction methods tested, since they overall gave quite similar results. CNN gave the best validation results for

all classification problems, however this was not followed up on the test results, which indicates that the CNN models overfitted. This makes it difficult to compare the results of texture features vs CNN, nonetheless the test results from CNN were quite on par with those archived from texture features for most of the classification problems.

5.3 Pre-processing

The skull stripped images performed better for all of the methods tested. The non smoothed skull stripped images performed better for almost all feature extraction methods, while for CNN the best validation accuracy models for all classification were split between smoothed and non-smoothed skull stripped images. However they results were quite similar between smoothing and non-smoothing, which likely was due to a relatively small smoothing kernel was used. The methods should also be tested with larger smoothing kernels, which may lead to more distinct differences one way or the other.

5.4 Limitations

Pre-processing

There was no manual inspection conducted for all images after normalization and segmentation of the brain, only a few images was inspected. This might have lead to some non-expected issues for some subjects, which might have influenced the results.

Feature extraction and Random Forest

No feature selection methods were used to reduce the number of features in this thesis. Some of the feature vectors had a very high dimension, and the use of a feature selection procedure might have improved the results and lead to better generalizability.

For RF only the total number of trees was varied, tuning of other parameters, e.g. max features per tree, could perhaps have lead to improved results.

Neural Network

For CNN there was a significant difference between the validation and test accuracy. The reason for this can be that be that no cross validation for neural network was conducted, and the models with the best validation result were presented with the results from the test set. Another reason can be that early stopping combined with dropout, might cause the final network model to be overfitted to the validation data, and thus performs relatively poorly on the test data. This also makes it difficult to conclude on

which of the hyper parameters for the neural network which works best for the task at hand, since the models with the best validation accuracy likely are overfitted.

Other

In this thesis the classes in the classification problems were made equal by undersampling the majority group(s), other methods for assuring equal instances of the classes should be investigated, such as synthetic minority oversampling or by adjusting for the cost for the overrepresenting class. This would have lead to more training data and likely improved the results.

For all of the method tested in this thesis, there were not done any form of correction for age, which might have improved the performance since since some of the biomarkers, e.g. atrophy could also be a symptom of normal aging, and the persons age is something that is known at the time of diagnostics.

In this thesis there is only extracted features from the whole brain, and the results could possibly have improved by focused on some defined ROIs, e.g. hippocampus, as previously done by Sorensen et al. [3].

5.5 Conclusion

In this thesis it was investigated if early diagnosis of AD could be reliably identified using MRI of the brain together with either 3D texture features and RF or 3D CNN. The data set used consisted of 2688 MRI scans from 811 subjects, and was obtained from the ADNI. The best accuracy results of the four-class early differential diagnosis problem was 41.1% and for the two-class problem discriminating between MCIs and MCIc, was only 60.3%.

For the other classification problems, which are more common in the research literature the best accuracy results achieved were 48.9% for NC vs MCI vs AD, 82.5% for NC vs AD, 67.6% for NC vs MCI and 67.9% for MCI vs AD. This is in lower end of the comparable research studies.

5.6 Future Work

For future work there would be interesting to see if the results can be improved by using data augmentation to generate more training data. This can also be combined with the use of pre-training of CNN, e.g. with the use of auto-encoders or sparse auto-encoders, which may reduce the risk of overfitting.

For future work there is also recommended to use more standardized data sets, data selection and pre-processing pipelines to make research more reproducible and comparable. Samper-Gonzalez et al. provides some interesting thoughts on this matter [35]. If public available data sets such as ADNI, AIBL or Oasis is used, Clinica[1] is a possible software platform to use, as it provides standardized pipelines for data selection and pre-processing. Clinica had a issue regarding ADNI data at the time pre-processing was done in this thesis, but that issue should now be resolved.

[1] http://www.clinica.run/

Bibliography

[1] M. Prince, A. Comas-Herrera, M. Knapp, M. Guerchet, and M. Karagiannidou, "World alzheimer report 2016: improving healthcare for people living with dementia: coverage, quality and costs now and in the future," 2016.

[2] "2017 alzheimer's disease facts and figures," *Alzheimer's & Dementia: The Journal of the Alzheimer's Association*, vol. 13, pp. 325–373, April 2017.

[3] L. Sørensen, C. Igel, A. Pai, I. Balas, C. Anker, M. Lillholm, M. Nielsen, A. D. N. Initiative, *et al.*, "Differential diagnosis of mild cognitive impairment and alzheimer's disease using structural mri cortical thickness, hippocampal shape, hippocampal texture, and volumetry," *NeuroImage: Clinical*, vol. 13, pp. 470–482, 2017.

[4] K. Oppedal, K. Engan, T. Eftestøl, M. Beyer, and D. Aarsland, "Classifying alzheimer's disease, lewy body dementia, and normal controls using 3d texture analysis in magnetic resonance images," *Biomedical Signal Processing and Control*, vol. 33, pp. 19–29, 2017.

[5] A. Payan and G. Montana, "Predicting alzheimer's disease: a neuroimaging study with 3d convolutional neural networks," *arXiv preprint arXiv:1502.02506*, 2015.

[6] E. E. Bron, M. Smits, W. M. Van Der Flier, H. Vrenken, F. Barkhof, P. Scheltens, J. M. Papma, R. M. Steketee, C. M. Orellana, R. Meijboom, *et al.*, "Standardized evaluation of algorithms for computer-aided diagnosis of dementia based on structural mri: the caddementia challenge," *NeuroImage*, vol. 111, pp. 562–579, 2015.

[7] S. Liu, S. Liu, W. Cai, S. Pujol, R. Kikinis, and D. Feng, "Early diagnosis of alzheimer's disease with deep learning," in *Biomedical Imaging (ISBI), 2014 IEEE 11th International Symposium on*, pp. 1015–1018, IEEE, 2014.

[8] A. Sarica, A. Cerasa, A. Quattrone, and V. Calhoun, "A machine learning neuroimaging challenge for automated diagnosis of mild cognitive impairment." url: `https://www.kaggle.com/c/mci-prediction`, 2016. Accessed: 03.06.2018.

[9] I. Castiglioni, C. Salvatore, J. Ramirez, and J. M. G. Sáez, "Machine-learning neuroimaging challenge for automated diagnosis of mild cognitive impairment: Lessons learnt," *Journal of neuroscience methods*, 2018.

[10] A. Burns and S. Iliffe, "Alzheimer's disease," *BMJ*, vol. 338, 2009.

[11] D. P. Perl, "Neuropathology of alzheimer's disease," *Mount Sinai Journal of Medicine: A Journal of Translational and Personalized Medicine*, vol. 77, no. 1, pp. 32–42, 2010.

[12] C. R. Jack Jr, D. S. Knopman, W. J. Jagust, R. C. Petersen, M. W. Weiner, P. S. Aisen, L. M. Shaw, P. Vemuri, H. J. Wiste, S. D. Weigand, *et al.*, "Tracking pathophysiological processes in alzheimer's disease: an updated hypothetical model of dynamic biomarkers," *The Lancet Neurology*, vol. 12, no. 2, pp. 207–216, 2013.

[13] "Mild cognitive impairment." url: `https://www.alz.org/dementia/mild-cognitive-impairment-mci.asp`. Accessed: 01.06.2018.

[14] A. J. Mitchell and M. Shiri-Feshki, "Rate of progression of mild cognitive impairment to dementia–meta-analysis of 41 robust inception cohort studies," *Acta Psychiatrica Scandinavica*, vol. 119, no. 4, pp. 252–265, 2009.

[15] E.-W. Radue, M. Weigel, R. Wiest, and H. Urbach, "Introduction to magnetic resonance imaging for neurologists," *CONTINUUM: Lifelong Learning in Neurology*, vol. 22, no. 5, Neuroimaging, pp. 1379–1398, 2016.

[16] D. McRobbie, E. Moore, M. Graves, and M. Prince, *MRI from picture to proton*. Cambridge University Press, January 2006.

[17] T. Ojala, M. Pietikäinen, and D. Harwood, "A comparative study of texture measures with classification based on featured distributions," *Pattern Recognition*, vol. 29, no. 1, pp. 51–59, 1996.

[18] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, pp. 971–987, Jul 2002.

[19] G. Zhao and M. Pietikainen, "Dynamic texture recognition using local binary patterns with an application to facial expressions," *IEEE transactions on pattern analysis and machine intelligence*, vol. 29, no. 6, pp. 915–928, 2007.

[20] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1, pp. 886–893 vol. 1, June 2005.

[21] R. Dupre and V. Argyriou, "3d voxel hog and risk estimation," in *2015 IEEE International Conference on Digital Signal Processing (DSP)*, pp. 482–486, July 2015.

[22] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[23] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*. John Wiley & Sons, Inc, second ed., 2001.

[24] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[25] J. Nagi, F. Ducatelle, G. A. Di Caro, D. Cireşan, U. Meier, A. Giusti, F. Nagi, J. Schmidhuber, and L. M. Gambardella, "Max-pooling convolutional neural networks for vision-based hand gesture recognition," in *Signal and Image Processing Applications (ICSIPA), 2011 IEEE International Conference on*, pp. 342–347, IEEE, 2011.

[26] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, 2014.

[27] M. W. Weiner, P. S. Aisen, C. R. Jack, W. J. Jagust, J. Q. Trojanowski, L. Shaw, A. J. Saykin, J. C. Morris, N. Cairns, L. A. Beckett, *et al.*, "The alzheimer's disease neuroimaging initiative: progress report and future plans," *Alzheimer's & dementia: the journal of the Alzheimer's Association*, vol. 6, no. 3, pp. 202–211, 2010.

[28] "About adni." url: `http://adni.loni.usc.edu/about/`. Accessed: 09.05.2018.

[29] B. T. Wyman, D. J. Harvey, K. Crawford, M. A. Bernstein, O. Carmichael, P. E. Cole, P. K. Crane, C. Decarli, N. C. Fox, J. L. Gunter, D. Hill, R. J. Killiany, C. Pachai, A. J. Schwarz, N. Schuff, M. L. Senjem, J. Suhy, P. M. Thompson, M. Weiner, and C. R. Jack, "Standardization of analysis sets for reporting results from adni mri data," *Alzheimer's Dementia: The Journal of the Alzheimer's Association*, vol. 9, pp. 332–337, May 2013.

[30] "Mri pre-processing." url: `http://adni.loni.usc.edu/methods/mri-analysis/mri-pre-processing/`. Accessed: 03.05.2018.

[31] "The mni brain and the talairach atlas." url: `http://www.nil.wustl.edu/labs/kevin/man/answers/mnispace.html`. Accessed: 02.05.2018.

[32] P. K. Mandal, R. Mahajan, and I. D. Dinov, "Structural brain atlases: design, rationale, and applications in normal and pathological cohorts," *Journal of Alzheimer's Disease*, vol. 31, no. s3, pp. S169–S188, 2012.

[33] R. Dupre, V. Argyriou, D. Greenhill, and G. Tzimiropoulos, "A 3d scene analysis framework and descriptors for risk evaluation," in *2015 International Conference on 3D Vision*, pp. 100–108, Oct 2015.

[34] X. Wang, T. X. Han, and S. Yan, "An hog-lbp human detector with partial occlusion handling," in *Computer Vision, 2009 IEEE 12th International Conference on*, pp. 32–39, IEEE, 2009.

[35] J. Samper-Gonzalez, N. Burgos, S. Fontanella, H. Bertin, M.-O. Habert, S. Durrleman, T. Evgeniou, O. Colliot, A. D. N. Initiative, *et al.*, "Yet another adni machine learning paper? paving the way towards fully-reproducible research on classification of alzheimer's disease," in *International Workshop on Machine Learning in Medical Imaging*, pp. 53–60, Springer, 2017.

Appendix A

Additional Validation Results

A.1 LBP features

Table A.1: Best LBP-TOP cells validation results from each pre-processing method, and classification problem.

	4-class	3-class	NC/ AD	NC/ MCI	MCI/ AD	MCIs/ MCIc
SS	**0.403**	**0.552**	**0.819**	**0.611**	**0.673**	**0.566**
	15, r3, 150	**15, all, 150**	**15, all, 75**	**15, all, 75**	**18, r1, 150**	**15, r1, 100**
sSS	0.377	0.544	0.802	0.600	0.620	0.561
	15, r3, 100	15, all, 150	15, all, 100	15, all, 100	15, r2, 100	16, r3, 30
WM	0.332	0.469	0.707	0.600	0.620	0.542
	15, r3, 100	15, all, 150	18, all, 150	15, all, 150	15, r1, 50	18, r2, 50
sWM	0.356	0.505	0.754	0.563	0.636	0.536
	15, r2, 100	18, r1, 150	18, r3, 150	18, r1, 150	15, r1, 75	18, all, 75
MNI	0.352	0.490	0.725	0.555	0.644	0.535
	15, all, 30	18, r3, 75	15, r3, 75	15, all, 50	18, all, 75	18, r3, 75
sMNI	0.349	0.498	0.736	0.562	0.628	0.535
	15, r3, 150	15, r3, 75	15, r2, 150	18, r3, 100	15, all, 150	18, r3, 30

4-class: NC/ MCIs/ MCIc/ AD, 3-class: NC/ MCI/ AD, SS: Skull Stripped, WM: White Matter, MNI: MNI normalized, s: smoothed. All results are given by the mean of the 10-fold cross validation, with the parameters in LBP-TOP-cells below, cell size, radius (r1: r1 & p8, r2: (r2 & p12), r3: (r3 & p16)) and number of trees in random forest. The best validation result from each classification problem are in bold.

A.2 3D VHOG features

Table A.2: Best 3D VHOG features validation results from each pre-processing method, and classification problem.

	4-class	3-class	NC/ AD	NC/ MCI	MCI/ AD	MCIs/ MCIc
SS	0.390	**0.548**	**0.811**	**0.609**	**0.671**	0.577
	15, 2, 50	**15, 2, 100**	**18, 2, 75**	**15, 2, 150**	**15, 2, 75**	18, 1, 50
sSS	**0.392**	0.527	0.798	0.600	0.646	**0.597**
	15, 2, 100	18, 2, 75	15, 2, 100	15, 2, 100	15, 2, 75	**15, 2, 50**
WM	0.341	0.495	0.729	0.583	0.641	0.530
	18, 2, 150	18, 2, 100	18, 2, 75	15, 2, 100	15, 1, 75	18, 2, 150
sWM	0.360	0.501	0.767	0.601	0.611	0.543
	15, 2, 100	18, 2, 100	18, 2, 100	15, 2, 150	18, 2, 150	18, 1, 30
MNI	0.362	0.511	0.766	0.582	0.622	0.548
	15, 2, 100	18, 2, 75	18, 2, 150	15, 2, 100	18, 2, 100	15, 2, 100
sMNI	0.353	0.481	0.774	0.611	0.611	0.530
	15, 2, 150	18, 2, 75	18, 2, 100	15, 2, 100	18, 2, 75	18, 2, 75

4-class: NC/ MCIs/ MCIc/ AD, 3-class: NC/ MCI/ AD, SS: Skull Stripped, WM: White Matter, MNI: MNI normalized, s: smoothed. All results are given by the mean of the 10-fold cross validation, with the parameters in 3D VHOG below, cell size, block size in cells and number of trees in random forest. The best validation result from each classification problem are in bold.

A.3 LBP - HOG features

Table A.3: Best LBP-TOP cells and 3D HOG combined validation results from each pre-processing method, and classification problem.

	4-class	3-class	NC/ AD	NC/ MCI	MCI/ AD	MCIs/ MCIc
SS	**0.396**	**0.555**	**0.815**	**0.612**	**0.670**	**0.576**
	18, r1, 150	**15, all, 150**	**15, r1, 50**	**15, r2, 150**	**15, r1, 100**	**18, all, 75**
sSS	0.385	0.542	0.804	0.601	0.666	0.574
	15, r3, 75	15, r2, 100	15, r2, 150	18, r2, 150	15, all, 75	15, r1, 30
WM	0.338	0.484	0.714	0.597	0.636	0.534
	15, r3, 75	18, all, 150	18, r3, 50	18, all, 100	18, r3, 75	18, r3, 100
sWM	0.358	0.495	0.746	0.588	0.636	0.530
	18, all, 100	18, all, 150	18, all, 75	15, r2, 100	15, all, 150	18, r3, 10
MNI	0.346	0.505	0.735	0.579	0.639	0.521
	15, all, 150	15, all, 100	18, r3, 150	18, r1, 150	15, r3, 100	18, r3, 150
sMNI	0.363	0.503	0.735	0.577	0.624	0.544
	15, r2, 150	15, all, 50	15, all, 100	18, r3, 75	15, all, 150	15, r2, 10

4-class: NC/ MCIs/ MCIc/ AD, 3-class: NC/ MCI/ AD, SS: Skull Stripped, WM: White Matter, MNI: MNI normalized, s: smoothed. All results are given by the mean of the 10-fold cross validation, with the parameters in LBP-TOP cells and 3D VHOG combined below, cell size, radius (r1: r1 & p8, r2: (r2 & p12), r3: (r3 & p16)) and number of trees in random forest. The best validation result from each classification problem are in bold.

A.4 Neural Network

Table A.4: Best CNN validation results from each pre-processing method, and classification problem.

	4-class	3-class	NC/ AD	NC/ MCI	MCI/ AD	MCIs/ MCIc
SS	0.375	0.592	**0.891**	0.656	**0.786**	0.667
	6, 0.1	6, 0.2	**6, 0.2**	6, 0.2	**6, 0.2**	6, 0.1
sSS	**0.448**	**0.615**	0.836	**0.729**	0.750	**0.787**
	6, 0.1	**6, 0.1**	4, 0.2	**6, 0.2**	5, 0.1	**4, 0.1**
WM	0.388	0.567	0.781	0.633	0.740	0.675
	3, 0.2	6, 0.1	4, 0.2	4, 0.1	5, 0.2	6, 0.2
sWM	0.394	0.567	0.820	0.642	0.698	0.725
	4, 0.2	4, 0.1	5, 0.2	5, 0.0	5, 0.2	6, 0.2
MNI	0.365	0.558	0.781	0.708	0.708	0.708
	6, 0.2	4, 0.2	5, 0.1	6, 0.0	4, 0.0	6, 0.1
sMNI	0.427	0.500	0.789	0.625	0.729	0.625
	5, 0.0	5, 0.2	4, 0.1	6, 0.2	4, 0.2	6, 0.1

4-class: NC/ MCIs/ MCIc/ AD, 3-class: NC/ MCI/ AD, SS: Skull Stripped, WM: White Matter, MNI: MNI normalized, s: smoothed. All results are given by the validation result, with the number of convolution layers and dropout below. The best validation result from each classification problem are in bold.

Appendix B

Program files

The program files is available here: 

B.1 Set-Up and feature extraction

B.1.1 Set-Up (PYTHON)

create_csv.py

Combines several meta files from ADNI, and gives each subject a class code, 1 for NC, 2 for MCIs, 3 for MCIc, 4 for MCIo and 5 for AD.

The produced csv files **baseline_meta.csv** and **extended_no_change.csv** are also attached.

All the ADNI standardized sets from ADNI must be downloaded. This includes:

- ADNI1:screening 1.5T
- ADNI1:Complete 1Yr 1.5T
- ADNI1:Complete 2Yr 1.5T
- ADNI1:Complete 3Yr 1.5T
- ADNI1:Annual 2Yr 1.5T

And the following csv files must be download from ADNI:

- ADNI_ScreeningList_8_22_12.csv
- ADNI_Complete1YearVisitList_8_22_12.csv
- ADNI_Complete2YearVisitList_8_22_12.csv
- ADNI_CompleteAnnual2YearVisitList_8_22_12.csv

- ADNI_CompleteVisitList_8_22_12.csv
- ADNI_ScreeningList_8_22_12.csv
- PTDEMOG.csv
- ADNIMERGE.csv
- DXSUM_PDXCONV_ADNIALL.csv

Pre-processing

For the following programs to work, SPM12[1] must be downloaded.The MRI from ADNI must be pre-processed using SPM12, and the MRI filenames given a prefix for the pre-processing step.

B.2 Feature extraction (MATLAB)

LBP-TOP

The LBP-TOP matlab program is not attached. It must be download from `http://www.cse.oulu.fi/CMV/Downloads/LBPMatlab`, and some modifications are necessary for LBP-TOP cells and the combination LBP-TOP and 3D VHOG to work. The LBP-TOP image needs to be saved in addition to the histograms.

hog3d.m

The hog 3d [21] is attached with the licence. Some modifications are made.

Extra functions for the matlab programs

Some functions needed for the feature extraction functions is in the folder: non-main-m-files.

TA_main.m

Specifies the parameters for features extraction, and finds the filenames of all the MRI images used to extract features.

lbp_cells_combined.m

Creates LBP-TOP cells feature from the LBP-TOP images, and combines the features from LBP-TOP and 3D VHOG.

[1] http://www.fil.ion.ucl.ac.uk/spm/software/spm12/

B.3 Classification (PYTHON)

B.3.1 Random Forest

classify_features.py

The main file for random forest. Sets which classification problem and number of trees to be tested.

create_npz_dataset.py

Loads the .mat features produced in feature extraction and splits into training and test set. The training set further split into x-fold-cross-validation sets, and saves it into .npz files.

classify_functions.py

Some functions functions for the random forest classification.

B.3.2 Convolutional neural networks

ann_main.py

Main file for creation and training of CNN. Sets the hyperparamters for the networks, data set, log directory etc.

ann_construct_models.py

Trains of the networks specified from ann_main, and saves the model and produces tensorboard files.

div_function.py

Contains various functions that is used for training of the neural networks.