University of Stavanger

FACULTY OF SCIENCE AND TECHNOLOGY

MASTER'S THESIS

Study Program/Specialization: Information Technology – Automation and Signal Processing	Spring semester, 2018 Open / ~~Confidential~~
Author: Rune Bjerland Risanger	…… (signature author)
Instructor: Professor Trygve Eftestøl Supervisor: Postdoc Researcher Ketil Oppedal	
Title of Master Thesis: Dementia classification using deep learning and texture analysis methods on magnetic resonance images Norwegian Title: Demensklassifisering av magnetiske resonansbilder ved bruk av dyplæring og teksturanalyse	
ECTS: 30	
Subject Headings: Deep Learning, Support Vector Machines, Convolutional Neural Networks, Magnetic Resonance Images, Gray Level Co-occurrence Matrix, Principal Component Analysis, Classification	Pages: 50 + Attachments/other: 5 Stavanger, 15th of June 2018

Dementia Classification Using Deep Learning and Texture Analysis Methods on Magnetic Resonance Images

Rune Bjerland Risanger

June 2018

Master's Thesis

Faculty of Technology and Science
Department of Electrical Engineering and Computer Science
University of Stavanger

Supervisors
Professor Trygve Eftestøl
Postdoc Researcher Ketil Oppedal

Abstract

Dementia is becoming an increasingly severe economical and socio-economical disease, as treatment is problematic, with different disease variants difficult to prevent and delay. With a rise in life expectancy, further problematic increase is expected to occur. Clinical diagnosis is difficult, with poor inter- and intra-rating between doctors. Developing tools for computer assisted diagnosis (CAD) for diagnosis verification could provide benefits for doctors and patients alike.

The primary objective of this thesis is to develop a CAD-system, to be implemented on T1-weighted magnetic resonance images (MRI) of normal controls (NC) and patients with either Alzheimer's dementia (AD) or Dementia with Lewy Bodies (DLB). Comparing results achieved through deep learning (DL) with texture analysis (TA) techniques together with Support Vector Machines (SVM) were also of importance. The CAD system was developed as a differential diagnosis system including all three groups in one classifier, but all binary classifications were also evaluated.

Results on a dataset of 760 subjects do not directly suggest if either method outperforms the other, with an achieved total accuracy of 66 % and 59 % for CNN- and SVM-classification respectively. Prior comparable studies have reported overall better accuracies on more shallow datasets, with results in this thesis suffering less on account of potential over-fitting issues. Limitations for DL-classification include dataset size and amount of evaluated architectures. The dataset could be expanded through availability of more study data or exploration of several data augmentation methods. Other potential limitations include lack of additional MR sequences or other modalities such as PET scans, with additional features possibly generating better results for the SVM-classifier. An enlarged dataset and additional TA methods could yield enhanced performance for CNN- and SVM-classifiers respectively.

Preface

The thesis was written at the Department of Electrical Engineering and Computer Science at University of Stavanger, during the spring semester of 2018. I would like to thank my supervisors, Professor Trygve Eftestøl and Postdoc Researcher Ketil Oppedal for their advice, feedback and backing. I'm most grateful for your dedication. I would also direct my sincere appreciation to my friends and family for support during this period.

Contents

Abbreviations

CAD	Computer Assisted Diagnosis
TA	Texture Analysis
AD	Alzheimer's Disease
DLB	Dementia with Lewy Bodies
NC	Normal Controls
MRI	Magnetic Resonance Image
GLCM	Gray level co-occurence matrix
DL	Deep Learning
SVM	Support Vector Machine
ANN	Artificial Neural Network
CNN	Convolutional Neural Network
SPM	Statistical Parametric Mapping
GM	Gray Matter
WM	White Matter
CSF	Cerebrospinal Fluid
PCA	Principal Component Analysis
ROI	Region of Interest
ADNI	Alzheimer's Disease Neuroimaging Initiative
MNI152	Standard-space T1-weighted Average Structural Template Image

Chapter 1

Introduction

This chapter gives a general introduction, as well as motivation for performing the task. Thesis objective, thesis outline and introductory information are also covered.

1.1 Dementia

Dementia is a general term for a decline in mental ability that interferes with a person's ability to perform daily activities [1]. Diagnosis can be demanding and MRI can provide non-invasive methods for boosting prediction accuracy [2]. Between the years 2000 and 2013, amount of deaths caused by cardiac arrest, stroke and prostate cancer has been reduced by 14 %, 23 %, and 11 % respectively. During the same period, the amount of deaths caused by AD has grown by 71 % [3]. The increased number of diagnosed dementia patients is a growing concern in today's world, and a rise in life expectancy is expected to further these problems. This has led dementia to become a severe economical- and socio-economical disease, as treatment is both expensive and problematic to handle. The total estimated worldwide cost of dementia is 818 billion US dollars in 2015, which represents 1.09 % of global GDP. By 2018, the global cost of dementia will rise above one trillion US dollars [4]. Average per-person Medicare-related payments for services to patients over 65 years with AD and other dementias are more than two and a half times as great as payments for all people without these conditions, with Medicaid payments 19 times as great [3]. Symptoms of dementia can vary from person to person, but a diagnosis is given when there are cognitive or behavioral symptoms that include[1]:

- Interfere with ability to function at work or at usual activities.
- Represent a decline from previous levels of functioning and performing.
- Not explained by delirium or major psychiatric disorder.
- The cognitive or behavioral impairment involves a minimum of two of the following domains.
  - Impaired ability to acquire and remember new information.
  - Impaired reasoning and handling of complex tasks, poor judgement.
  - Impaired visuospatial abilities.
  - Impaired language function.
  - Changes in personality, behavior or comportment.

1.1.1 Alzheimer's disease

Alzheimer's disease is an irreversible, progressive neurological brain disorder that slowly destroys brain cells. AD causes short- and long term memory loss, and can eventually cause complete loss of ability to accomplish most activities and tasks. Degradation of neurons in brain cells is assumed to be related to the formation of amyloid plaques and neurofibrillary tangles [5]. The cognitive decline caused by AD ultimately leads to dementia [6].

Alzheimer's Disease is the most common cause of dementia, estimated to be between 60 and 80 percent of cases [4] and, as of 2016, is estimated to have infected over 44 million people worldwide. The amount of diagnosed dementia patients is expected to double every 20th year on average [4]. As of 2016 an estimated 1-in-4 people with AD gets diagnosed, and receive necessary treatment.

1.1.2 Dementia with Lewy Bodies

Dementia with Lewy bodies is a type of progressive dementia that is caused by abnormal microscopic deposits that damage brain cells over time. DLB is the second most common type of degenerative dementia in patients older than 65 year [7], after AD. DLB is distinguished from other types of dementias by the presence of parkinsonism, neuroleptic sensitivity, fluctuations in consciousness,

and spontaneous visual hallucinations. The combinations and severity of symptoms varies from patient to patient [8]. The presence of α-synuclein is primarily in neurodegenerative disorders like Parkinson Disease and DLB, but is found secondarily in AD too [9].

The true frequency of DLB compared to other types of dementia remains unclear, with previous studies reporting a prevalence range from zero to 22.8 % of all dementia cases. A different study reported a 4.2 % occurence of all diagnosed dementias in the community. In secondary care, the amount was noted to increase to 7.5 % [10]. The reported values are probable underestimates, as the three studies that focused on identifying DLB, and included a neurological examination, showed a significantly larger proportion (16–24 %) [7].

1.2 Deep Learning in Neuroimaging

Deep learning algorithms, CNNs in particular, have established themselves as popular choices for analyzing medical images [11]. The algorithms have been reported to improve previous state of the art classification accuracy by more than 30 % in several multidimensional areas, including speech-, image-, video- and text-recognition. Prior state of the art methods were reported to struggle to obtain more than 1–2 % improvements [12]. These promising results led to its implementations in neuroimaging, which has provided encouraging results, due to the unique characteristics of medical images [13]. One of its main upsides compared to other classifiers is the automatic feature learning, which removes a level of subjevtivity from feature extraction, and is believed to be the main contribution to improvements in accuracy. Previous results show that deep learning methods are able to learn physiologically important representations and detect latent relations in neuroimaging data [12]. The algorithms has provided promising results for both feature extraction and classification, being able do extract patterns outside general techniques. Previous study results have reported that machine learning algorithms can predict AD more accurately than an experienced clinician [14].

1.3 Thesis Objective

Primary objective of this thesis is to develop a CAD system to be implemented on T1-weighted MRIs of healthy patients, and patients with AD and DLB. Evaluating DL-classification in a neuroimaging problem, compared to that of standard SVM-classification based on TA, is also a priority. While resulting classifiers should manage a three class problem directly, being able to discriminate between each of the viable two class problems also carry great promise.

A classifier that can reliably separate healthy patients from patients with AD or DLB could prove helpful in early detection of diseases. There are no cures for any type of dementia as of today, but with early and correct diagnosis, several benefits can be achieved. It has been shown that early detection and intervention at its prodromal stage, are effective in delaying the onset of dementia [15]. When detected at an early stage, patients can be helped to remain at an acceptable mental condition for a longer period, behavioural changes can be easier managed, and symptom progression can be slowed down [2]. Early diagnosis can also help relieve families of several stressful situations, and help patients live as well as possible with the disease. A reliable tool for classification could also diminish the amount of people affected by the disease without receiving the appropriate diagnosis. The problem at hand focuses on differential diagnosis, but reliable feature learning could provide benefits for early detection of diseases.

Correct diagnosis of patients is also of great importance, as AD- and DLB patients can behave differently and respond differently to medication. However, it's possible for patients to have symptoms of more than one dementia related disease at the same time. Currently, only one method for differentiating AD and DLB exist, the dopamine transporter scan. It's an expensive piece of equipment which can't be made available at all centres [2]. However, co-morbidity is a factor within subjects with dementia. Subjects with co-morbidity have one labelled true state of nature when it comes to classification purposes, but might have hallmarks of several diseases.

Slowing down the dementia process is a challenging scenario today, but the disease has gathered attention in the medical community for its increasing problems. With no cure available, early detection for slowing down the progressive nature of the disease is important, but new treatment methods are being studied. Constructing tools that can assist doctors in making early and correct diagnosis of the disease can provide potential gain, both economically and for compassionate reasons. As of now, there are only clinical diagnosis of the diseases,

meaning the doctors make a calculated guess based on MRI images, and grade of mental function reduction. Developed classifiers of brain MRI can provide helpful tools for doctors when performing the diagnosis.

1.4 Thesis Outline

Chapter 2 - Background

This chapter outlines the background for the thesis and theory behind implemented methods.

Chapter 3 - Materials and methods

This chapter describes implementation of the aforementioned methods. Experimental set-ups for DL- and SVM-classification are covered, as well as dataset generation.

Chapter 4 - Results

This chapter presents the achieved results for the experiments covered in the previous chapter.

Chapter 5 - Discussion

The results and limitations are discussed in detail in this chapter.

Chapter 6 - Conclusion

The final conclusions of the thesis are presented. Possible improvements and recommendations for future work will also be included.

Chapter 2

Background

This chapter provides an overview of background theory applied in this thesis. Dataset origin and methods for classification and pre-processing are also covered.

2.1 Magnetic Resonance Imaging

Magnetic Resonance Imaging is a method for producing non-invasive accurate anatomical brain representations [16]. These scans yield 3D volumes representing the brain, and can be of high resolution while offering good contrast between different brain tissues. There exists both Functional Magnetic Resonance Imaging (fMRI) and structural MRI, where T1-weighted and T2-weighted structural images are widely used. T1-weighted images excel at contrast between Gray Matter (GM) and White Matter (WM), while T2-weighted images separate Cerebrospinal Fluid (CSF) from GM and WM. The solid contrast between GM and WM for T1-weighted scans makes MRI a superior choice for investigation of diseases that affect the central nervous system [17]. The brain volumes can be split into a series of coronal-, sagittal- and axial slices, as visualized in figure 2.1.1.

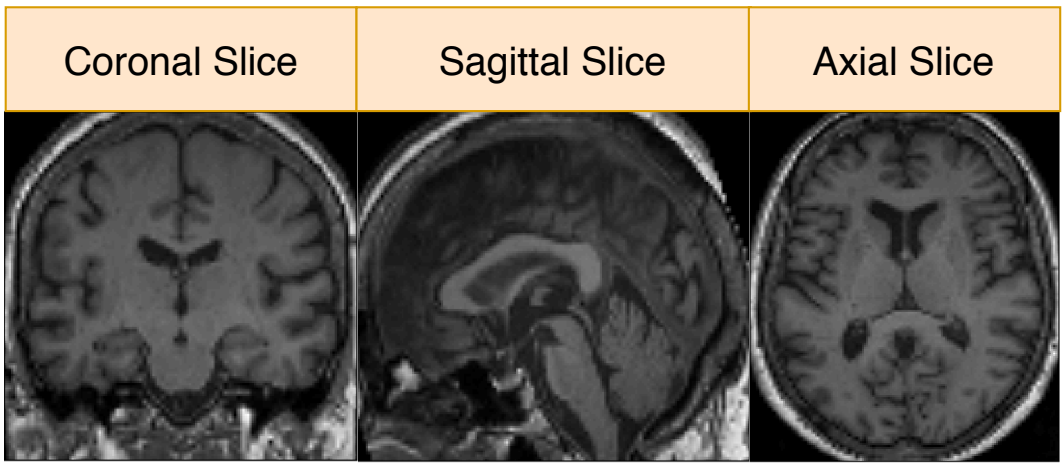

Figure 2.1.1: A T1 structural image can be split into a series of coronal-, sagittal- and axial images, as visualized in the figure. Each 3D volume has an image resolution with a certain depth, given in the x-, y- and z-direction

EDLB study

The DLB-consortium aims to establish guidelines for the clinical diagnosis of DLB and establish a common framework for the assessment and characterization of pathologic lesions at autopsy [18]. Substantial progress has been made in regards to the detection and recognition of DLB as a common and important clinical disorder [19].

ADNI study

The ADNI study is a global research effort that actively supports the investigation and development of treatments that slow or stop the progression of AD[1,2].

2.2 Pre-processing MRI

This section presents a pre-processing approach for the dataset.

Several factors influence classifier performance in neuroimaging. One vital factor is similar and proper pre-processing of data, which is valid for both CNN- and SVM-classification. The performance of CNN are largely affected by input data. Comparable pre-processing of the dataset is of huge importance for classifiers

[1]Data used in preparation of this article were obtained from the Alzheimer's Disease Neuroimaging Initiative (ADNI) database (adni.loni.usc.edu). As such, the investigators within the ADNI contributed to the design and implementation of ADNI and/or provided data but did not participate in analysis or writing of this report. A complete listing of ADNI investigators can be found at:

[2]ADNI Acknowledgement List

and their potential results. MRIs consist of important brain tissues, but the skull, eye-sockets and extracerebral tissues[1] are also prevalent in the scans. These areas are undesirable when extracting features related to dementia diseases, and can be considered noise factors, preferably removed without affecting crucial information.

As data within the ADNI- and EDLB-study were obtained at different locations with varying equipment and pre-processing, variations were present in the dataset, and required identical pre-processing to limit unnecessary variations. For DL classification, data has to be of same size, and can be achieved through resizing[2].

Several pre-processing methods have been constructed to limit factors not relevant to the diseases, with the ones used in this project being covered in this chapter[3]. It must be noted that all forms of pre-processing has, according to the no free lunch theorem [21], drawbacks in addition to their benefits.

2.2.1 Spatial Normalization

With size and relative brain position between subject essential for identifying sparse features, implementing a method for template mapping was essential. By executing spatial normalization, the volumetric images are generalized to a common template, sharing an identical coordinate system. The MNI152 template was used for this thesis, and was generated by averaging 152 anatomical scans after correcting for overall brain size and orientation [22].

Normalization will limit translational differences and size abnormalities in particular, highlighting structural differences in the brain instead. However, as volumetric data is stretched towards a template, slight information loss is inevitable.

2.2.2 Brain tissue segmentation

Volumetric images of the brain possess a lot of information, where specific tissues are decisive for recognizing dementia. GM and WM can possess information relevant to brain-related diseases, including different types of dementia.

[1] Non brain related tissues, including skin and eyeballs-

[2] For the avid reader, spatial pyramid pooling [20] is a method developed for handling diverse input sizes.

[3] Due to only having structural T1-images available for all subjects, realigning for motion correction and co-registering structural and functional MRI-images has not been included in the preprocessing pipeline.

There's typically some cerebral atrophy[1] happening with age, with AD patients often suffering to a greater extent. Extracting information hidden in form and size of brain tissue could provide good features for differentiating AD with DLB or NC.

Accentuating these relevant tissues include segmenting the brain to dispose of noise factors. There are several available tools for brain segmentation, with Freesurfer [23] and Statistical Parametric Mapping (SPM) [24] being examples. The volumetric data for each subject was segmented into 6 parts; GM, WM, CSF, extracerebral tissues, the skull and the surroundings. With non-brain tissue providing information with diminishing returns for dementia recognition, removing them without damaging relevant information is essential.

Skull stripping

Whole-brain segmentation, termed skull stripping, is a crucial technique for the analysis of neuroimaging data [25]. Noise factors are segmented and discarded from brain tissue to avoid unnecessary features in the image dataset. Skull stripping is a thresholding technique used on brain tissue segments, reconstructing brain volumes without its noisy counterparts.

2.2.3 Smoothing

Some form of smoothing is usually introduced when performing image classification. By smoothing volumetric data with a low pass filter, high frequency artefacts can be removed from the image, which improves signal-to-noise-ratio (SNR). Spatial normalization removes most translational artefacts between subjects, but modelling errors occur, with voxel to voxel mapping not being perfect as a result.

Image smoothing shares voxel information to outlying voxels, shifting information to its surroundings. Improved overlap of corresponding areas between subjects can be achieved. A neural network relies on spatial information being in comparable areas for different subjects, as to identify different sparse features between classes. Translational differences prevalent in vectorized neural network inputs can harm classifier performance significantly, by feature learning becoming more challenging.

For TA, smoothing is a double edged sword. All pre-processing methods have unfavorable ramifications associated with them. Smoothing lowers spatial res-

[1] Brain cell size decrement, which can be explored in GM- and WM-structure.

olution, resulting in information loss. Prior to CNN-classification, 3D volumes are flattened to a vectorized feature space[1], making spatial information loss relatively insignificant. Feature extraction obtains features concealed in volumetric data, making spatial information essential for attaining satisfying results. Smoothing before TA has created debate in neuroimaging, as SNR improves with diminishing spatial information. Other undesirable side effects includes a partial-voluming artefact along the edges of the brain, where brain voxels become smoothed with no-brain voxels [26], and a similar artefact between GM- and WM tissue.

Smoothing generally concludes the pre-processing pipeline, as its effects are undesirable prior to other pre-processing methods.

2.3 Texture analysis

This section presents texture analysis and the method used for obtaining feature vectors for all subjects.

Through TA a different layer of information is attainable from brain volumes [27], which refers to information attained from an image's appearance, structure and arrangement.

Throughout this thesis, comparing DL-classification with feature based SVM-classification is the priority. Performing texture analysis aims to extract statistical features from brain volumes, and adopt these features for class differentiating. Statistical learning methods are a valuable tool for decoding information from neural imaging data [28].

2.3.1 Gray level co-occurrence matrix

The gray level co-occurence matrix is a statistical tool used for image classification [29], that makes extraction of statistical information from pixel distribution possible [27]. Pixel distribution analysis can be performed for several distances and directions, extracting different layers of information concealed in images.

Matrices can be derived directly from an image's original pixel values, but a grouping of comparable pixel values are regularly used, as better information is generally secured. For an 8 bit image of pixel values ranging from 0-255, 8 or 16

[1] After CNN feature extraction, which can handle multidimensional data.

groups are frequently used. The method declares how often grayscale groups appear alongside each other, for specified directions and distances.

In 2D and 3D space there are 4 and 13 unique directions, with each pixel connecting to 8 and 26 pixels respectively. The remaining directions are covered by the distance parameter, which can appear both positive and negative. Performing the method for multiple directions together might give smoother results, but subtle features in images can be missed. The GLCM is computed pixel by pixel for the entire image, with the resulting matrix adopted for statistical analysis. Figure 2.3.1 visualizes the method for a 2D image.

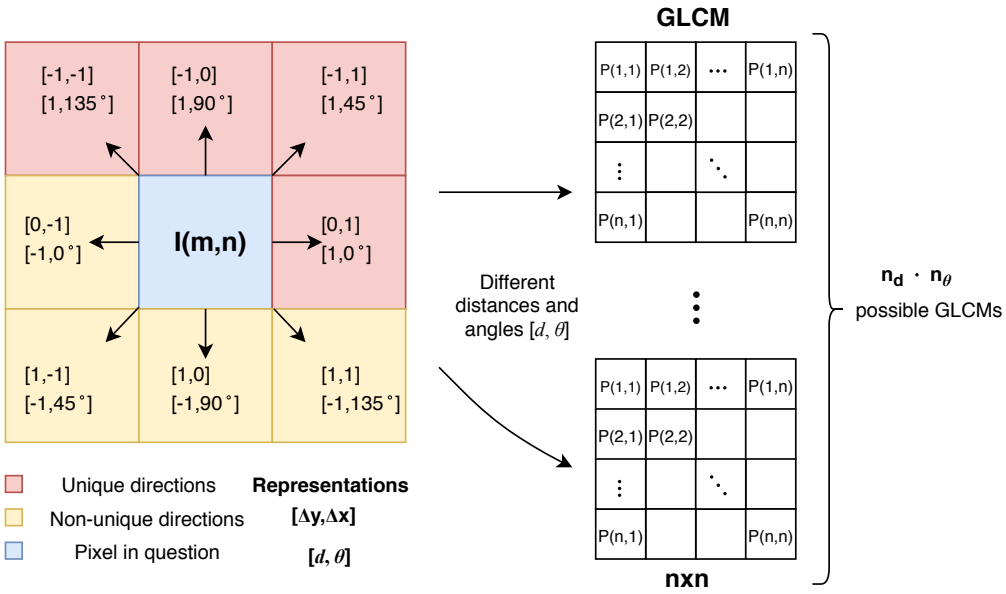

Figure 2.3.1: The figure visualizes GLCM for a 2D matrix, which is a pixel by pixel operation that reports relative frequency of similar pixel values appearing alongside each other, with a distance d and angle θ. The number of rows and columns in the matrices represent the amount of grayscale groups. To get valuable information from the GLCM, a graylevel image of 256 unique pixel values is often grouped into 8 or 16 pixel value groups.

Statistical Analysis

Brain tissue structure might provide valuable information regarding a subject's brain, and if a form of dementia is present. Performing statistical analysis on brain volume pixel distributions can provide valuable features for discriminating various dementia types and healthy controls. Up to 22 statistical features can be extracted from pixel value distributions of an image, including energy, entropy, contrast, variance and correlation [30].

2.4 Support Vector Machines

This section presents Support Vector Machines, the procedure used for feature vector based classification for all subjects.

SVMs are supervised learning models used for classification or regression analysis. Such a classifier introduces hyperplanes to separate labelled data in feature space. Intuitively, acceptable partioning can be achieved by a hyperplane that achieves the largest distance towards the nearest training datapoint of any given class. The larger the distance, the lower generealization error is expected [31]. The hyperplane is adjusted as to minimize expected error rate, given in equation 2.4.1.

$$J(\mathbf{w}) = \frac{1}{2}\left\|\mathbf{w}\right\| + C\sum_{i=1}^{N}\zeta_i \tag{2.4.1}$$

The expected error rate is subject to the constraints of equation 2.4.2.

$$y_i\left(\mathbf{w}^T\Phi(x_i) + b\right) \geq 1 - \zeta_i \quad \text{and} \quad \zeta_i \geq 0, \quad \text{for} \quad i = 1, \cdots, N_{\text{classes}} \tag{2.4.2}$$

Where C penalizes the error for *i* classes. **w** is the vector of coefficients, with b representing parameters for handling non-separable data. The Φ kernel is used to transform data from input to feature space. Errors are penalized more with larger C, compared to that of a lower value. Adjustment of the preceding parameter is done as to avoid either under- or over-fitting the model.

Classification is performed by introducing data unknown to the classifier, with classification determined based on which side of the hyperplane data lands in feature space. Figure 2.4.1 shows a feature space of two different classes, where different values of C has been visualized.

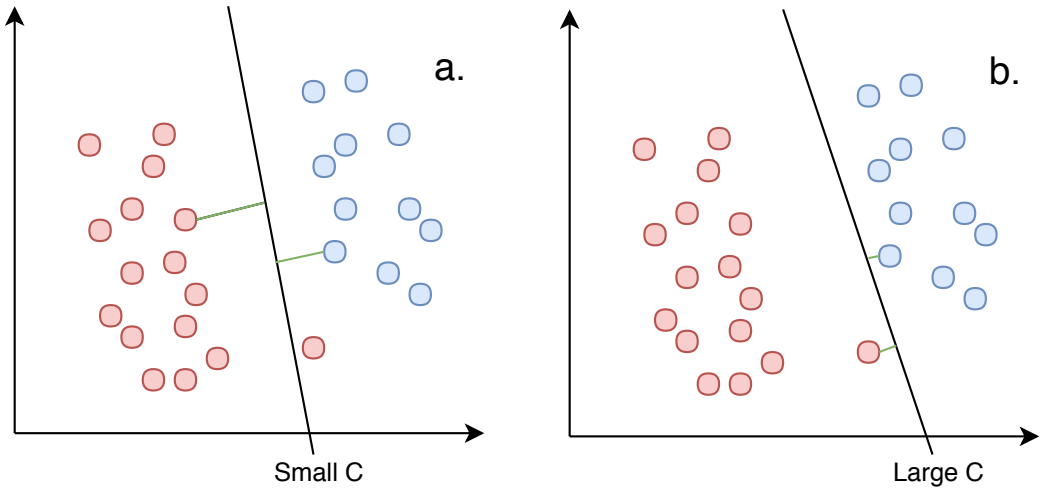

Figure 2.4.1: A visualization of a hyperplane separating two classes with different values of C. C works as a regularization parameter for the SVM-classifier. **a** - A meagre C-value will focus on maximizing the minimum margin, visualized with the green line. **b** - A substantial C-value focuses on a hyperplane that correctly classifies as many training samples as possible. A balanced value is generally needed to avoid both under- and over-fitting, as both constraints are rarely satisfied simultaneously

2.5 Principal Component Analysis

This section describes Principal Component Analysis (PCA), which was used for feature vector reduction.

PCA is a procedure that identifies relationships between objects and is widely used for data reduction. The operation convert a set of observations of possibly correlated variables into a set of uncorrelated principal components. PCA is a viable choice for data reduction, when sets of features are expected to correlate strongly with each other. It's defined as an orthogonal, linear transformation that remodel data to a new coordinate system, with the resulting coordinate system projectioning the greatest variance representation available in the data at its first principal component. The procedure generates the second principal component with its current best variance represantation, and so on [32]. Intuitively, it seeks a linear combination of variables such that the maximum variance is extracted from the variables. Equation 2.5.1 shows how a data vector from the original space is transformed into a space with **L** principal components.

$$\mathbf{T_L} = \mathbf{XW_L} \tag{2.5.1}$$

Where the **T**-vector represent the transformed and reduced form of **X**, through the loading vector **W**[1].

A covariance matrix presents how N variables correlate with each other. To find the principal components, the eigenvalues with its corresponding eigenvectors is computed from the covariance matrix. The eigenvalue reflect the quality of the projection to a lower number of dimensions, with a higher value including more data variance and a better data representation. Figure 2.5.1 shows how an example of dimensionality reduction with PCA, with two features being reduced to one.

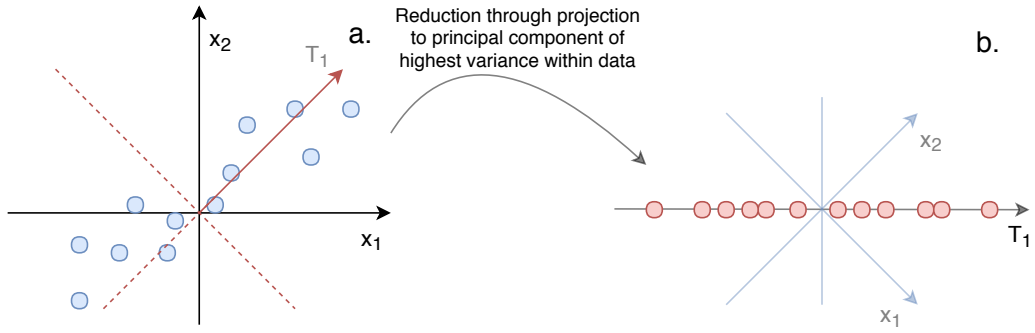

Figure 2.5.1: PCA is visualized, with one principal components, T_1 being used. **a** - Initial feature space of x_1 and x_2. **b** - Culminating feature space of T_1, which best describes variations in given data.

2.6 Neural Networks

This section introduces Artificial Neural Networks (ANN), their building blocks and theory behind them.

2.6.1 Artificial Neural Networks

In machine learning, artificial neural networks are models for approximating mathematical algorithms. Networks are used for learning complex problems, are able to handle multi-dimensional problems and develop non-linear models. They are loosely based on the human brain, and are constructed to mimic its learning process. The building blocks of the networks are artificial neurons, which are based off of biological neurons. An artificial neuron is shown in figure 2.6.1

[1]Dimensionality: $\mathbf{T}_{NxL}$ $\mathbf{X}_{Nxp}$ $\mathbf{W}_{pxL}$

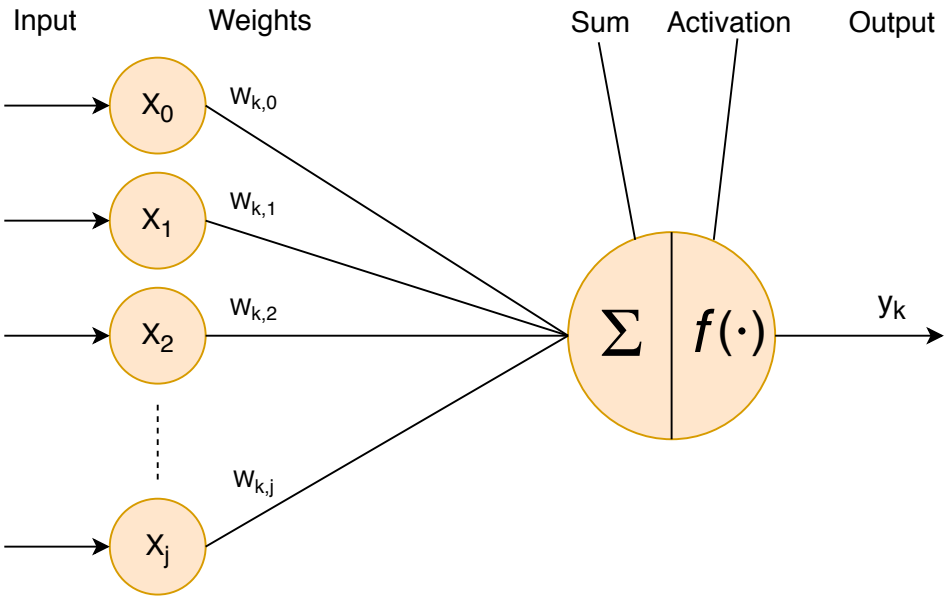

Figure 2.6.1: An artificial neuron receives a weighted sum of inputs, which is processed with an activation function for non-linearity. A neural network generally has hidden layers of several hidden nodes, where each hidden node refers to an artificial neuron. Each neuron has connections with unique weights associated with them.

The artificial neuron consist of the sum of several weighted inputs. The sum of weighted inputs is then affected by an activation function, yielding an output. Equation 2.6.1 shows the mathematical procedure done for every artificial neuron.

$$y_k = f\Big(\sum_{j=0}^{N} w_{kj} x_j\Big) \tag{2.6.1}$$

A conventional neural network consists of several layers of many artificial neurons, referred to as hidden layers of hidden units. When every hidden node from one layer is connected to every hidden node of a different layer, each connection with its unique weight, a fully connected layer is established. These weights are adjusted when exposed to training data, as to learn the problem at hand. A neural network with one hidden fully connected layer is shown in figure 2.6.2.

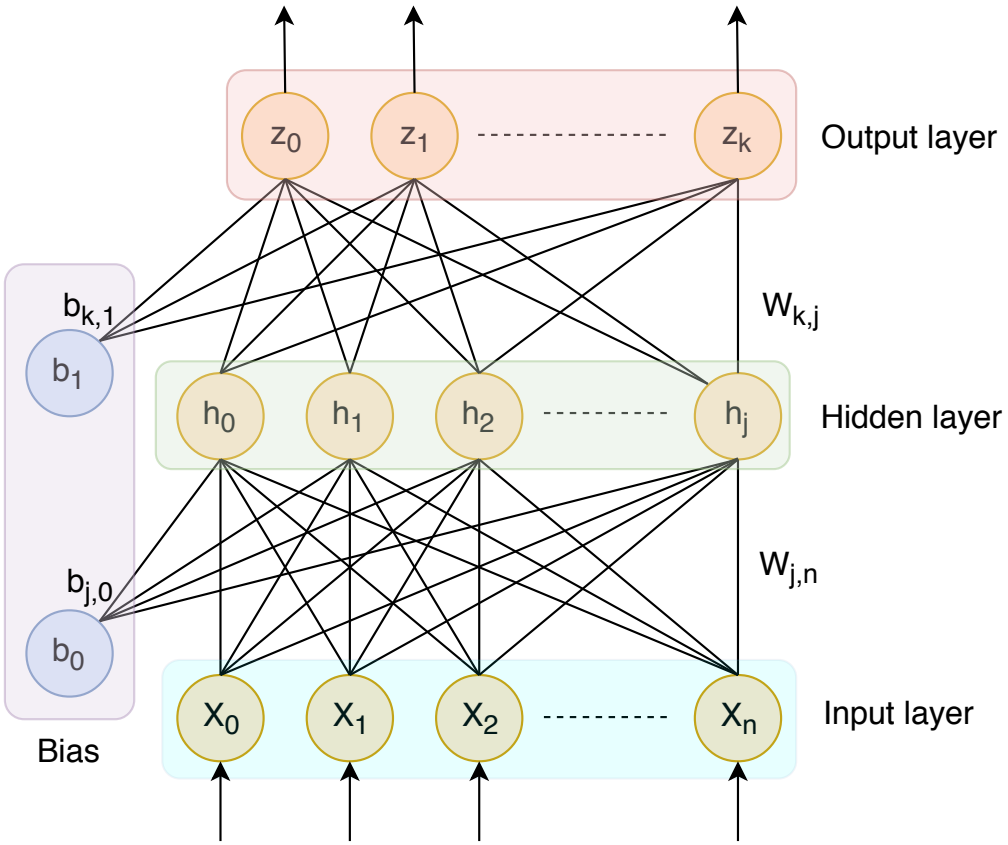

Figure 2.6.2: A general layout for a feed forward neural network. Such a network consists of an input layer, one or more hidden layers and an output layer. Every connection has a unique weight associated with it

There exists countless tasks that a normal person would class as simple, but can be hard to explain algorithmically. Fitting examples are various forms of text-recognition, including hand-writing of letters, words and numbers. While numbers or letters can be easily interpreted among people, describing a letter based directly on a perceived visual experience is not quite as simple. While our brain can adjust to subtle variations in people's handwriting and appearance of letters or numbers, the same can't be said directly for a computer. An ANN's performance is directly related with its exposure to data variation of the impending problem. Related to the previous example, an ANN would require large amounts of training data, with variations within the training samples. With enough variation present in training data, a network can reliably uncover enough differences to be able to distinguish different numbers, letters or similar problems.

To build upon the previous example, an ANN could have problems recognizing variations in handwriting if its exposure to training data consisted entirely of the writing pattern of a single person. Furthermore, this implies that high amounts of exposure to similar training data can lead to over-fitting, which refers to large

performance gaps between training data and other data. Similarly, networks of complex architectures can construct more intricate algorithms for separating data, establishing decision boundaries shaped after exposed training data, which is not necessarily a representative for true data distribution. A simplified case of different levels of complexity of a decision border algorithms is shows in figure 2.6.3

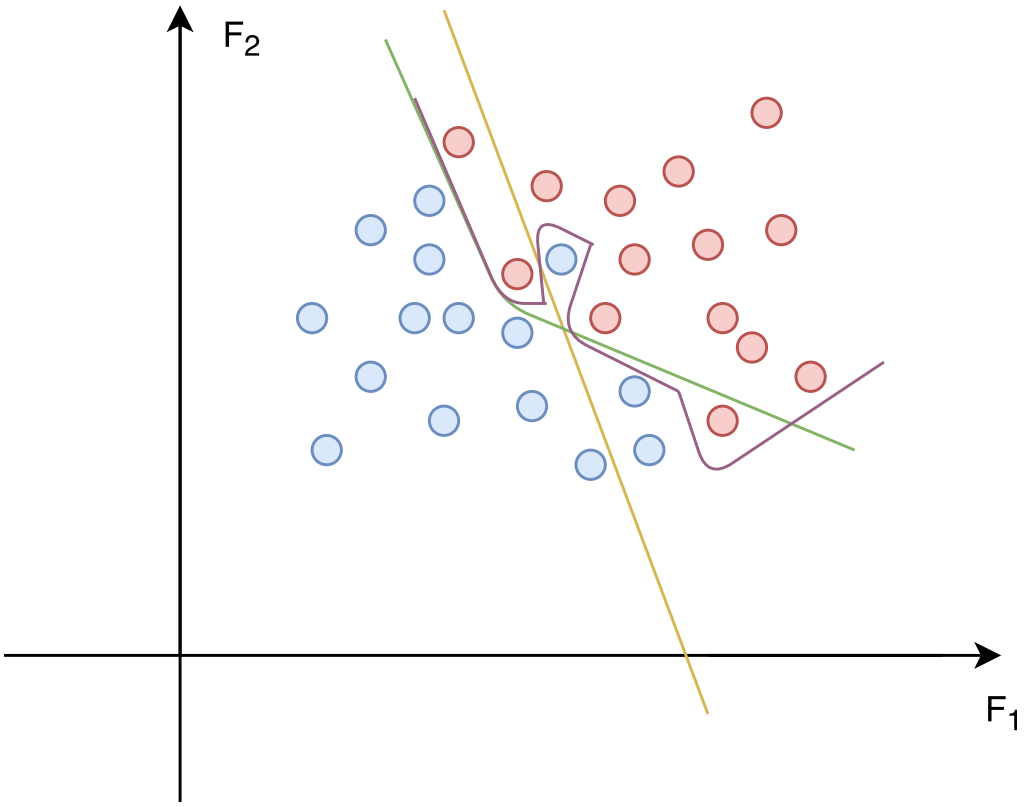

Figure 2.6.3: Different complexity of algorithms is visualized. Over-fitting leads to decision boundaries designed directly after training data, and not representing the true distribution of data, making it perform worse when introduced to new data.

As demonstrated in figure 2.6.3, optimal decision border algorithms represents true data distribution, and are not formed directly after training data. Some level of complexity may be required to attain a classifier's optimum, but excessive design after training data generates classifiers that generalizes poorly.

2.6.2 Convolutional Neural Networks

This section introduces Convolutional Neural Networks, and the building blocks associated with them.

CNNs are a promising form of deep learning that specializes in multidimensional data. Implementation on problems of higher dimensionality like images, object- and speech-recognition, have provided promising results.

The fundamental difference of CNNs and a feed forward fully connected network occurs in layer connections. A fully connected layer has all hidden nodes of one layer connected to every hidden node of the previous layer, hence its name. In CNNs however, a hidden node connects only to a few close nodes of the preceding layer, subject to a set kernel size and stride. Furthermore, all units are connected to the previous layer in the same way, with the exact same weights and structure [33]. Figure 2.6.4 shows the difference between a convolutional- and a fully connected layer.

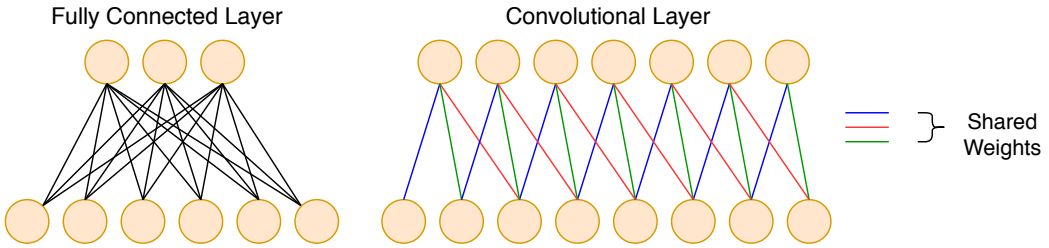

Figure 2.6.4: A representation of a fully connected layer, compared to that of a convolutional layer. A fully connected layer has all its hidden nodes of one layer connected to that the impending layer. A convolutional layer has only a few close nodes of the preceding layer connecting to a node of the impending layer, subject to a set kernel size and stride.

The strengths of the CNNs lies in their ability to extract features directly and perform directly at multidimensional data, whereas other neural networks require vector inputs. It's possible for a CNN to act as an encoder in front of a feedforward neural network, where the convolution output is vectorized by flattening.

Convolutional Layers

Convolutional layers are layers introduced to handle multidimensional data directly, and can extract features directly. While fully connected layers require vector inputs, convolutional layers can handle larger dimensions. Fully connected layers can handle images when reshaped to vector form, but spatial information is lost in this process. Brain volumes are 3D arrays that can be altered by exposure to convolutional layers, with characteristics concealed in brains volumes extracted to a feature space. Figure 2.6.5 visualizes 2D- and 3D-convolutions.

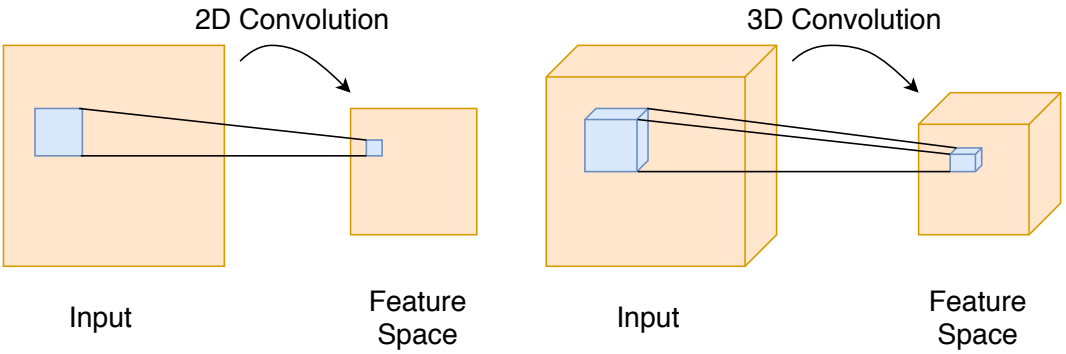

Figure 2.6.5: 2D and 3D convolutions is visualized. A kernel of set size is shifted over an input image, extracting its characteristics into a feature space.

Pooling layers

Pooling layers are layers introduced to down-sample input data. In neuroscience, brain volumes are of vast magnitude, making pooling layers essential for feature vector reduction. Down-sampling is introduced to avoid significant over-fitting, and reduce computational power needed, as 3D convolutions are monumental procedures. There exists several types of pooling options, where average-, weighted average- and max pooling are popular methods. Figure 2.6.6 shows an example of 2x2 max pooling.

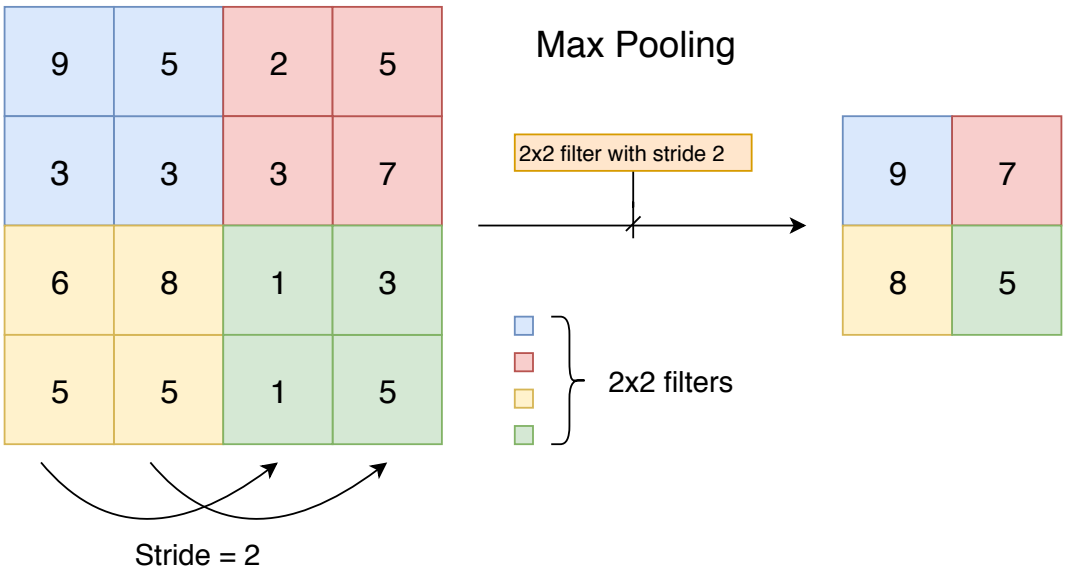

Figure 2.6.6: 2x2 max pooling for down-sampling is visualized. The resulting output will be $^1/_4$ of the input size, while preserving as much information as possible

2.6.3 Activation functions

This section describes activation function adopted for non-linear algorithms.

As observed in figure 2.6.1, the weighted sum of inputs is affected by an activation function prior to the generated output. Without implemented activation functions, the network would create severely limited algorithms, as non-linearity would be unattainable. The neuron triggers if the input to a node is significant, with the input altered by the given properties of the activation function. There are several activation functions being used in neural networks, which fits different purposes.

ReLU

An activation function that has received increased popularity over the last few years, is the ReLU function. Its increase in popularity has seen it overtake the sigmoid and tanh functions as the go-to activation function for hidden layers. Saturating non-linearities found in the tanh- and sigmoid activation functions are much slower than the non-saturating non-linearity of the ReLU function, when using variantions of gradient descent [34]. The ReLU function is generally recommended for activation in convolutional layers. Its equation is given in equation 2.6.2

$$f(\mathrm{x}) = \max(0, \mathrm{x}) \tag{2.6.2}$$

Because rectified linear units are nearly linear, they preserve many properties that make linear models easy to optimize with gradient-based methods. Properties that make linear models generalize well are also preserved[35].

ReLU6

A modification to the ReLU activation function is known as ReLU6, adding an output restriction between 0 and 6. ReLU6 has been stated as able to learn sparse features earlier [36]. Equation 2.6.3 shows the modification done to the original ReLU activation function 2.6.3.

$$f(\mathrm{x}) = \min\big(\max(0, \mathrm{x}), 6\big) \tag{2.6.3}$$

Softmax

The softmax activation function is generally implemented at the output layer of neural networks, generating outputs representing the probability of the out-

put belonging to each of the classes. The equation for the softmax is given in equation 2.6.4

$$f(\mathbf{z})_j = \frac{e^{z_j}}{\sum_{i=1}^{N} e^{z_i}} \quad \text{for} \quad j = 1, \cdots, N_{\text{classes}} \tag{2.6.4}$$

2.6.4 Back propagation

Managing complicated problems involves a neural network being fed training data during a training phase. Weights and biases are initialized with random values. By comparing the resulting output of the neural network with its target, the network will adjust its values to make better predictions for future training samples. The mean squared error (MSE) is regularly used when comparing the output with its corresponding target, working as a loss function for the weights **w**, and is shown in equation 2.6.5.

$$J(\mathbf{w}) = \frac{1}{2} \sum_{k=1}^{N} (t_k - z_k)^2 \tag{2.6.5}$$

The network's resulting output, given in the z-vector, is compared to the target vector for its true class, located in the corresponding t-vector. The t-vector is typically of one-hot format. An example of one-hot encoding for class number i is shown in equation 2.6.6

$$\mathbf{t} = [0, 0, 0, \cdots, 0] \quad \text{where} \quad t_i = 1 \quad \text{and} \quad \text{length}(\mathbf{t}) = N_{\text{classes}} \tag{2.6.6}$$

The term back-propagation is often misunderstood as the whole learning algorithm for neural networks. Back propagation refers to the method for computing the gradient, while a separate algorithm is used to utilize the aquired information for learning[35]. Popular algorithms include gradient descent, Adadelta [37] and Adam [38]. By the use of an optimization algorithm, the weights **w** are adjusted to minimise the MSE.

2.6.5 Hyper parameters

This section presents the hyper parameters used to adjust a network towards its optimum performance.

With architecture settled, adjustment of several hyper parameters is carried out, as to attain ideal performance for the given architecture. Hyper parameters adjusted during this thesis include *epochs*, *batch size*, *L2 regularization* (λ), *learning rate* (η) and *dropout*.

Epochs

The amount of epochs refer to the amount of times a full training set is applied to a network during a training phase. As weights are initialized randomly[1], completion of several epochs are anticipated to reach its potential, as weight adjustments happens gradually. However, a large amount of epochs can lead to the network adjusting excessively to the training data, yielding a classifier of low bias and high variance. Figure 2.6.7 visualizes the bias-variance tradeoff [39].

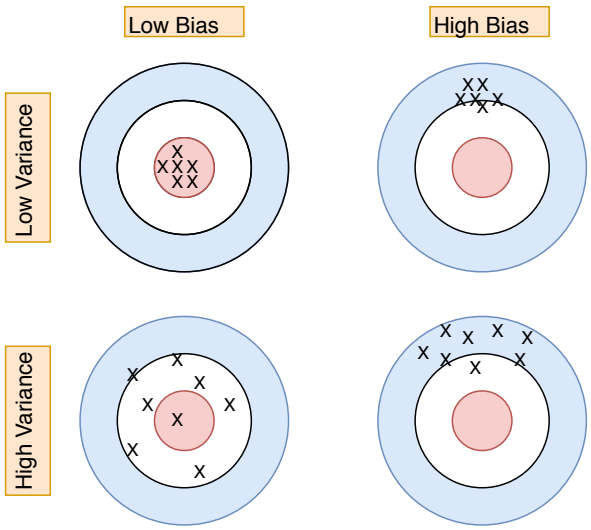

Figure 2.6.7: The bias-variance tradeoff visualizes outcome of required decisions made during the construction of a classifier. An under-fitted model with high bias will struggle to handle the complexity of a problem, while an over-fitted model will have problems generalizing, with its performance deteriorating when introduced to different data. A good classifier can handle data complexity, while still being able to generalize well.

Epoch abundance contributes to over-fitting, with a resulting network generally performing remarkably well on training data, with the network designed to recognize training data specifically. However, when exposed to data previously untouched by the network, performance is expected to deteriorate immensely.

Batch size

Batch size refers to the amount of training samples passed through the network

[1] If a pre-trained network is not used.

for each weight update. Its value can be set between 1 and the total number of training samples, with weights altered after every sample and once per epoch respectively. A lower batch size will result in further weight adjustments per epoch, compared to that of a higher batch size.

Learning rate

The learning rate controls how extensively weight adjustments are tuned with respect to the loss function. Equation 2.6.7 visualized the effect of the learning rate η, and how it affects the weight update through the weighted gradient of the error function.

$$\mathbf{w}_{i+1} = \mathbf{w}_i - \eta \cdot \frac{\partial}{\partial \mathbf{w}_i} J(\mathbf{w}_i) \tag{2.6.7}$$

Intuitively, the η-parameter specifies how quickly the weights are adjusted in the direction of the gradient, as shown in figure 2.6.8.

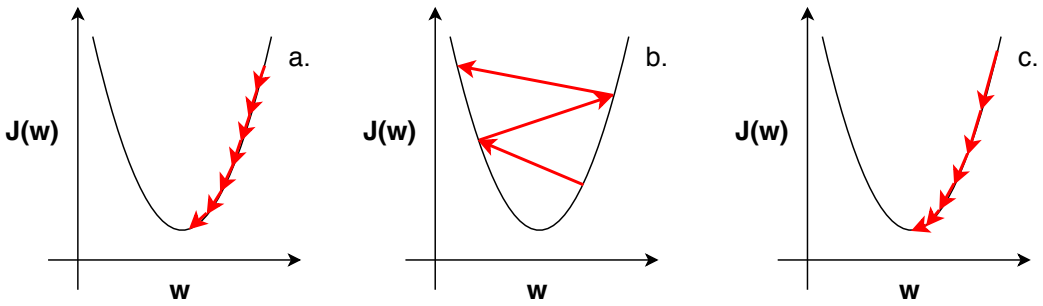

Figure 2.6.8: The figure visualizes weight adjustment based on different learning rates. **a** - Insufficient learning rate, too many iterations required to reach optimum. Can end up in local minima. **b** - Substantial learning rate, optimal weights can't be reached as they're adjusted too much per iteration. **c** - Practical learning rate, which can be adjusted over time. Sizeable learning rate at the start causes the gradient descent to avoid local minima and reach its imminent optimum, while a decrease over time ensures that optimal weights are achievable.

A substantial learning rate can have its corresponding loss function struggle at attaining saturation, with gradient descent of too vast increments. A meager learning rate on the other hand, while in theory able at attaining saturation, would require too many iterations to realistically reach it. Additionally, training with an insufficient learning rate might result in a local minima, rather than the global minima of the loss function. Figure 2.6.9 visualizes different values of η.

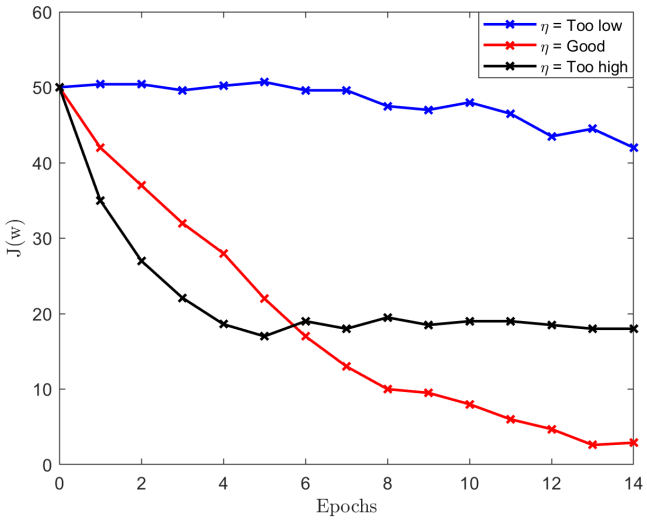

Figure 2.6.9: Different learning visualized in a plot. Large learning rates struggles at attaining saturation, as increments are too vast. An insufficient learning rate will be very slow and can result in local minimums, rather than the global minima for the loss function. A solid learning rate converges as swiftly as possible, while avoiding local minimums.

L2-regularization

Regularization penalizes complexity of learning models, reducing over-fitting [40]. A penalty for model complexity or extreme parameter values is added to the weight factors[1]. Equation 2.6.8 shows the inclusion of λ to the loss function given in 2.6.5.

$$J(\mathbf{w}) = \frac{1}{2}\sum_{k=1}^{N}(t_k - z_k)^2 + \frac{\lambda}{2}\sum_{i=1}^{M} w_i^2 \tag{2.6.8}$$

Excessive weights result in larger error, with the algorithm favouring modest weight factors. The term is incorporated in the weight gradient of the back-propagation term, with the gradient for hidden node connecting to the output node given in equation 2.6.9.

$$\Delta \mathbf{w}_{jk} = \eta \cdot \left[x_j \cdot (z_k - t_k) \cdot z_k \cdot (1 - z_k)\right] + \left[\lambda \cdot \mathbf{w}_{jk}\right] \tag{2.6.9}$$

Dropout

Dropout is introduced during training phase to counter over-fitting. By randomly dropping nodes and their connections during training phase, it prevents

[1] Does not include bias factors.

the network relying on a few monumental connection values [41]. Figure 2.6.10 shows an example of dropout implementation in a neural network.

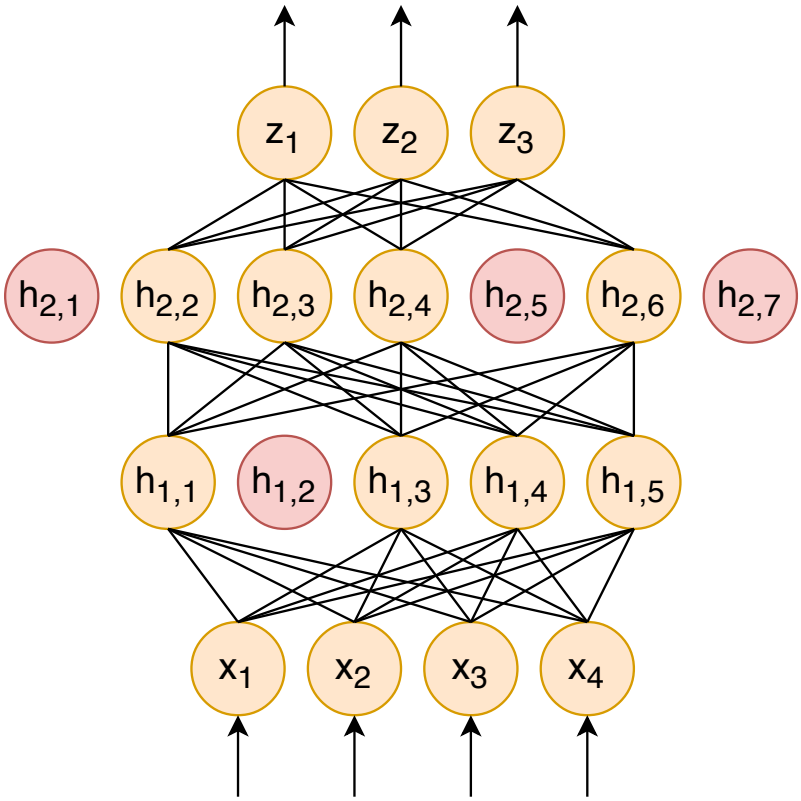

Figure 2.6.10: Dropout is introduced to counter over-fitting. Every single node has, during training phase, a probability of its connections being dropped associated with it. Dropout works as a regularizer, which hinders the network relying on a few colossal weights to perform classification

2.7 Confusion Matrix

This section presents the confusion matrix and the performance metrics derivable from it.

Visualizing supervised learning[1] classification results, involve the creation of a confusion matrix. Each row of the matrix represents the predicted class, while each column represents the actual class, or vice versa [42]. By organizing classification results in a confusion matrix, it's visualized where the algorithm misclassify, hence the name confusion matrix [43]. Figure 2.7.1 visualizes a confusion matrix for a three-class problem.

[1]The classification of pre-labelled data.

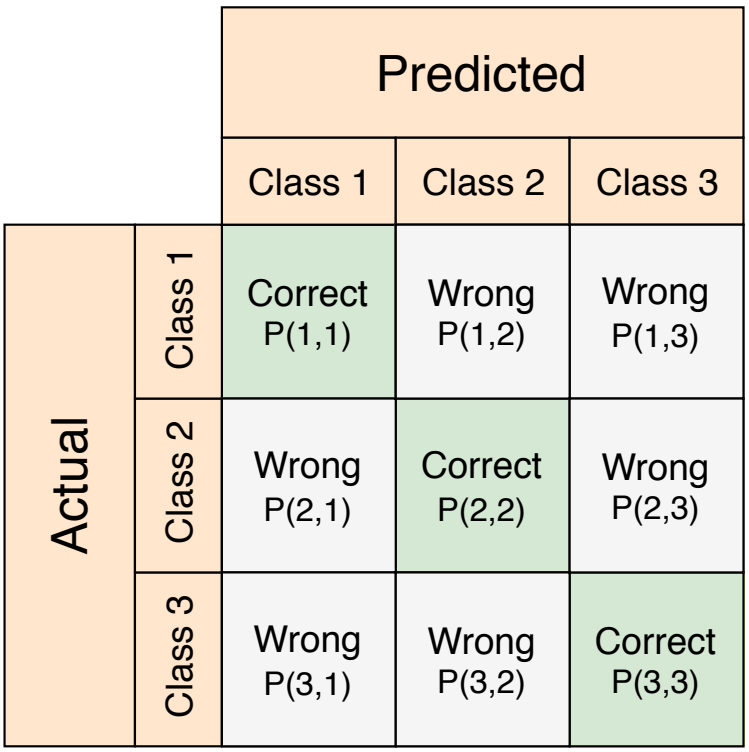

Figure 2.7.1: A confusion matrix visualizes where an algorithm misclassifies, hence the name. It's often used to calculate performance metrics to evaluate classifiers, including accuracy, precision and recall for every class.

2.7.1 Performance metrics

Several performance metrics can be derived from a confusion matrix, and can be used to evaluate classifier performance thoroughly.

Accuracy

Total accuracy refers to the proportion of total number of correctly predicted classes, and is an overall measure of classifier performance. Intuitively, accuracy can be regarded as the probability of correctly classifying a randomly selected sample. Accuracy is a well-documented performance metric for balanced datasets, but its utility decay with a skewed dataset [43]. Accuracy is calculated as shown in equation 2.7.1.

$$\text{TotAcc} = \frac{\sum_{i=1}^{N} P(i,i)}{\sum_{i=1}^{N}\sum_{j=1}^{N} P(i,j)} \tag{2.7.1}$$

Precision

Precision measures the proportion of predicted classes which is properly predicted within one class. The precision metric examines all values predicted of each class, and calculates how large proportion of these that are correctly predicted. Calculation of precision for class *i* is shown in equation 2.7.2.

$$\text{Prec}_\text{i} = \frac{\text{P(i,i)}}{\sum_{\text{j=1}}^{\text{N}} \text{P(j,i)}} \tag{2.7.2}$$

Recall

Recall is a measure of the proportion of a given class that is correctly predicted. Recall calculates the proportion of a given class that is correctly classified. Recall is calculated for class *i* according to equation 2.7.3.

$$\text{Rec}_\text{i} = \frac{\text{P(i,i)}}{\sum_{\text{j=1}}^{\text{N}} \text{P(i,j)}} \tag{2.7.3}$$

Precision and recall are calculated for each class to provide detailed analysis of individual class performance, as accuracy alone can grant misleading results, if datasets are unbalanced[1] [43]. For two-class problems, negative predictive value (NPV) and specificity are the equivalent of precision and recall for the second class. As the three class problem was an essential part of the project, NPV and specificity were included through calculating recall and precision for every class, visualized with the class-specific underscore given in equations 2.7.2 and 2.7.3. This was justified as to avoid confusion between performance metrics calculated for three class- and binary problems.

[1] If the amount of classes vary massively, the accuracy metric can be flawed. For representation, if a two-class problem has $^9/_{10}$ of its data belonging to a certain class, with the classifier subsequently predicting every sample to be of the aforementioned class, an accuracy of 90 % is achieved, even if the classifier is clearly unreliable.

Chapter 3

Materials and methods

This chapter explains the construction of the dataset, implementation of the pre-discussed methods on it, as well as the networks and its architectures. How the data is affected by different forms of pre-processing, as well as the types of classifiers and their structures, are also covered in this chapter.

3.1 Dataset construction

This section clarifies dataset construction and the reasoning behind decisions made.

The dataset used during this thesis consists of T1 structural MRI-images containing NC-, AD- and DLB scans. DLB-data was obtained from the EDLB-study, with AD- and NC-data supplemented through the ADNI-study. The data gathered from the different studies was analyzed before being added to the dataset. Severe outliers visually were discarded, as differences were of significant level. Outliers included brain volumes of shapes and size that varied greatly from the standard, and patients with confirmed co-morbidity. The amount of data from each class was kept at a relatively similar amount, as to not skew the dataset favourably towards certain classes. The dataset construction is shown in figure 3.1.1

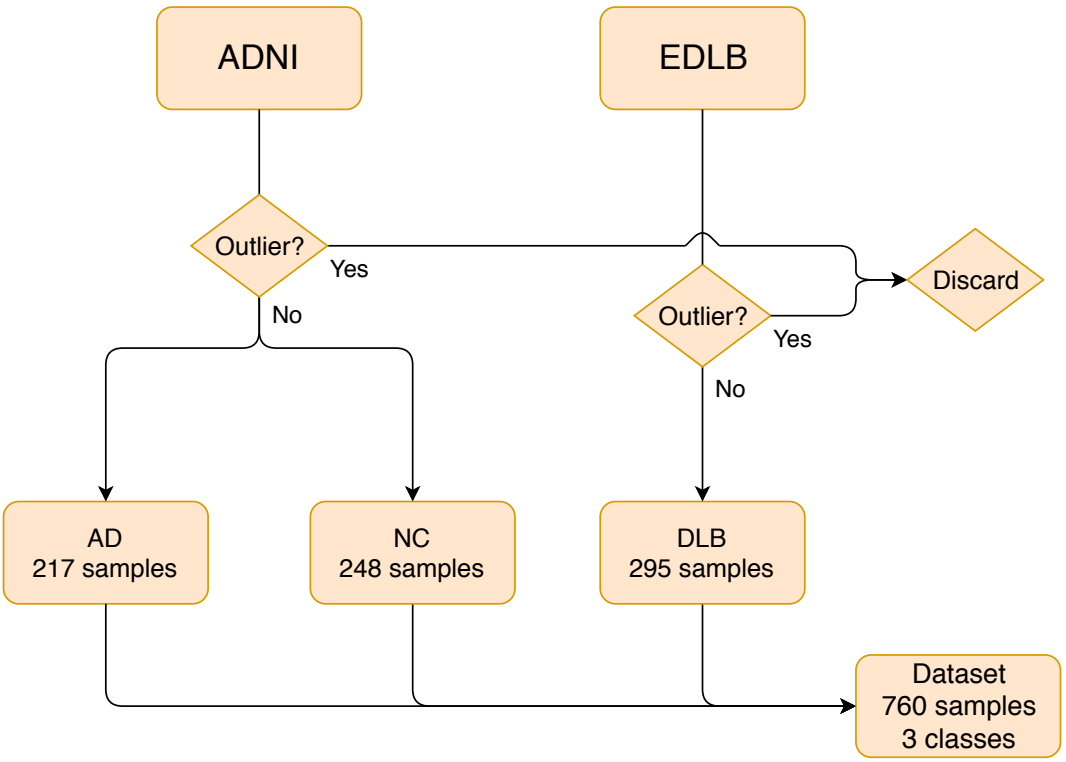

Figure 3.1.1: The dataset was constructed from baseline and screening data from the ADNI study, in addition to DLB data from the EDLB study. Severe outliers in the dataset were discarded.

For classification purposes, the dataset was split into sub groups. As the dataset was of significant size, a validation set was used alongside a training- and test set. Dataset distribution is visualized in figure 3.1.2.

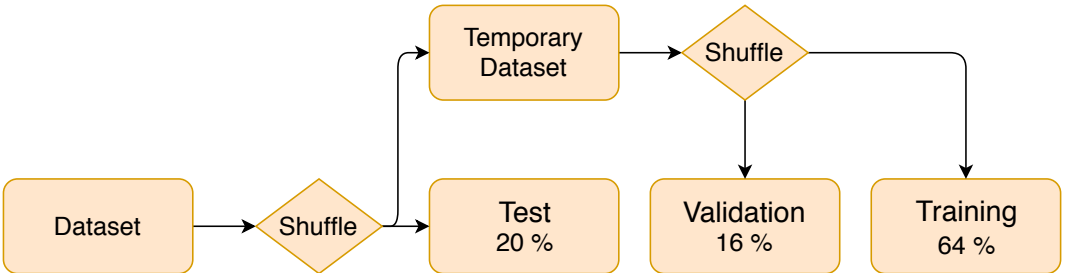

Figure 3.1.2: The dataset was split into three separate sets, a training-, validation- and test set. The classifiers was trained on the training data, with the validation set functioning as a regularization parameter. During CNN-classification, validation was performed after every epoch, as to notify the user when over-fitting was imminent. For SVM-classification, the parameters were adjusted to improve validation score. The test set was set aside as to provide an unbiased classification

The training set was used for training for each classifier. The C-parameter was tuned towards results on the validation set during SVM-classification. For the

CNN-classifier, validation was performed following each finished epoch, to regularize the classifier with early stopping[1]. For both classifiers, a test set was kept separate, as to provide unbiased classification at the end. Attaining comparable results included an identical split of training-, validation- and test set for both classifiers.

3.2 Pre-processing implementation

This section explains implementation of pre-processing methods introduced in section 2.2. Constructing a usable dataset involves all data undergoing similar pre-processing steps. The pre-processing pipeline used in the thesis is shown in figure 3.2.1.

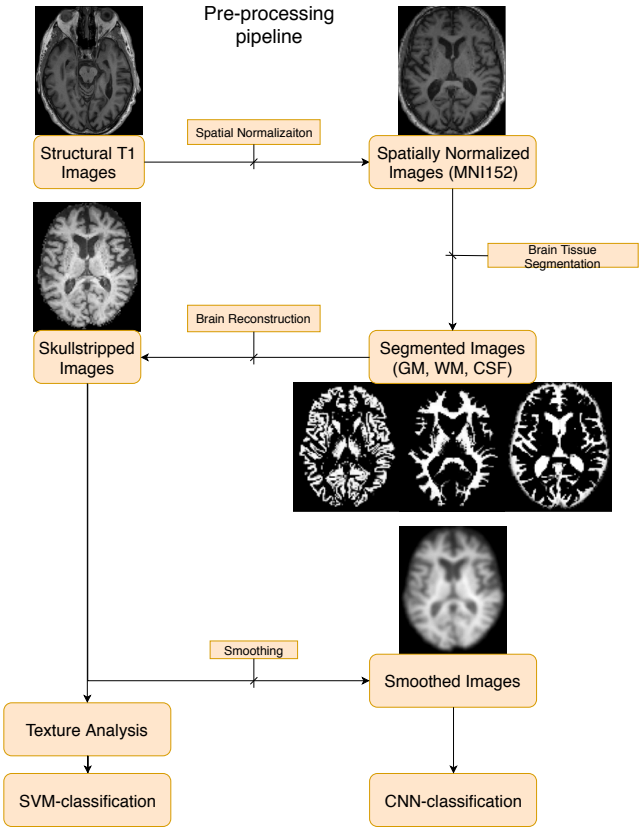

Figure 3.2.1: The pre-processing pipeline include spatial normalization-, skull stripped reconstruction through segmentation, with smoothing only performed for CNN-classification. The MNI152-template was used for normalization, with a combination of the original structural image and its GM-, WM and CSF-segments being used to create a skull-stripped brain volume. These volumes were used for TA, while the CNN had its image dataset smoothed before use.

[1] Early stopping refers to the halt of the classifier when the validation score has not improved after i epochs

By performing spatial normalization initially, the brain volumes are stretched towards a shared coordinate system, as to limit translational differences between subjects. Skull stripping by segmentation and reconstruction yields images where crucial features are retained, while worthless ones are abolished. TA adopted the resulting images directly, as to retain virtually all vital information. Smoothing is performed before CNN-classification to cancel out noise artefacts and improve SNR.

3.2.1 Spatial Normalization

When constructing a CNN for classifying datasets, a fully connected layer is used at the end of the network. While the previous layers consists of convolutional- and pooling layers for feature extraction and down sampling respectively, they won't directly classify input data. Resulting features can be introduced to a SVM-hyperplane for classification, but a fully connected output layer of size $N_{classes}$ is often used. With softmax activation as output, a direct probability representation for each class is given.

Input size will affect resulting output size, with inputs of varying size not working for networks with fully connected layers, as the amount of hidden nodes in theory would have to change for each iteration. When using inputs of varying size for a network with one or more fully connected layers, the feature maps will be of different size, resulting in inoperative architecture. As available data was gathered from several sources and studies, image resolution and depth varied between subjects. By implementing a fully connected layer at the end of a network, spatial normalization of brain volumes to identical size is mandatory. Data was mapped to a common 79z95x79 format, with resulting voxel size of 2x2x2 mm. Voxel volume was selected for maintaining high resolution brain volumes, while limiting computational power needed for time-consuming 3D-convolutions.

3.2.2 Brain tissue segmentation

The normalized volumetric images for each subject was segmented into GM, WM, CSF, extracerebral tissues, the skull and surroundings.

Skull stripping

The brain volumes were reconstructed as to maintain valuable information given in the brain tissues. By thresholding the sum of segmented brain tissues

and CSF, multiplied element by element with the structural T1 image, the brain volume was stripped of its noise factors, including the skull and extracerebreal tissues. This is in neuroimaging referred to as *skull stripping*. The equation used to skull strip the images is shown in equation 3.2.1.

$$\mathbf{i_1} . \cdot \left((\mathbf{i_2} + \mathbf{i_3} + \mathbf{i_4}) > 0.5 \right) \tag{3.2.1}$$

$\mathbf{i_1}$, $\mathbf{i_2}$, $\mathbf{i_3}$ and $\mathbf{i_4}$ represent the pre-segmented T1 volumetric image, the GM-segmented image, the WM-segmented image and the CSF-segmented image respectively. Skull stripping is vital when used for feature extraction or neural networks, as noise artefacts are discarded.

3.2.3 Smoothing

Several smoothing kernels were explored before all brain volumes were smoothed with a 5x5x5 kernel, before being fed to feature extraction through CNN.

3.3 Feature Extraction

This section describes implementation of feature extraction methods executed on the dataset.

As to generate feature vectors for SVM-classification, GLCM-matrices were computed with 13 unique directions and 4 different distances, being [1, 2, 4, 8] pixels, for all subjects. 22 different features were extracted through statistical analysis of resulting matrices for each subject, yielding a vector of in total 1144 features.

PCA

With feature vector size exceeding dataset size, PCA was implemented for dimensionality reduction. Several features were expected to be strongly correlated with each other, making PCA a valid choice for reduction, with resulting principal components being uncorrelated. PCA was implemented through scikit-learn [44], with principal components representing largest variance retained.

3.4 Experimental layout

This section describe the experimental layout for classifying with both the SVM- and CNN-classifier.

Desired purpose of the classifier involve classifying pre-processed MRI directly without prior knowledge. However, given prior knowledge, establishing classifiers able to reliably handle available binary problems could also aid in differential diagnosis scenarios. Given the significance of both the three class problem, and each of the available binary problems, classifiers were designed for managing all scenarios, each with their own CNN- and SVM-classifier.

3.4.1 SVM

For SVM-classification, feature vectors for each subject was introduced from TA. The dataset was normalized to attain zero mean and unit standard deviation. Feature normalization was based off of the training set, as to avoid contamination of the validation- and test set. SVM feature normalization is shown in equation 3.4.1.

$$\mathbf{z} = \frac{\mathbf{x} - \mu_{\mathbf{X}_{\text{Train}}}}{\sigma_{\mathbf{X}_{\text{Train}}}} \tag{3.4.1}$$

Dimensionality reduction was introduced through PCA, as to use a set of uncorrelated features. The SVM-boundary was modelled to fit the training data, with its parameters being adjusted based on its performance on the validation set. The classifier of best validation performance was then tested on the test set, as to provide an unbiased performance check. With improved test results, confusion matrices and performance metrics were generated to analyse results. Figure 3.4.1 shows the implemented SVM-pipeline.

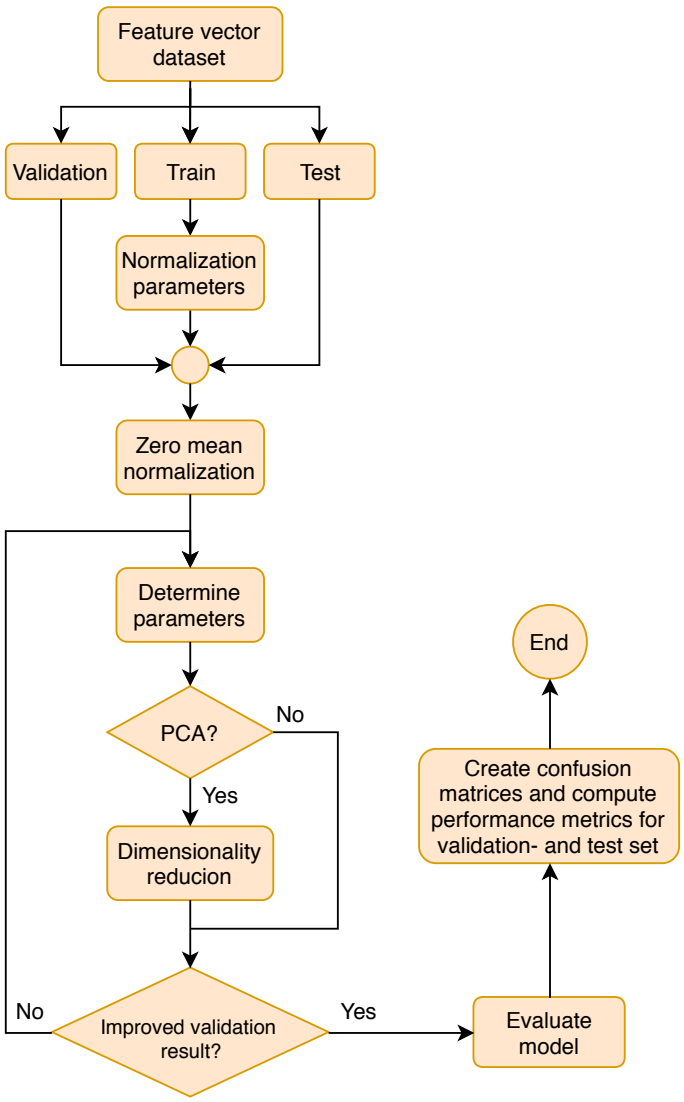

Figure 3.4.1: The SVM-pipeline include zero mean normalization based off of the training set. Parameters for the SVM-classifier were initialized and adjusted based on its performance on the validation set. Dimensionality reduction was implemented through PCA on the training set. To provide an unbiased performance check, the classifier was tested on a separate test at the end.

3.4.2 CNN

Normalization for the CNN-classifier was performed differently, as min-max normalization has been reported to outperform other normalization methods for data mining purposes [45]. Equation 3.4.2 shows the min-max normalization for the CNN dataset.

$$\mathbf{z} = \frac{\mathbf{x} - \min(\mathbf{x})}{\max(\mathbf{x})\text{-}\min(\mathbf{x})} \tag{3.4.2}$$

Architecture

Conventional CNN architecture for classification of multidimensional data consist of a combination of several convolutional-, pooling- and fully connected layers. Depth and width of the network are of great importance, as to not under- or over-fit. Deep and wide network results in complex algorithms, able to recognize data variations better, but can more easily lead to over-fitting. Shallow networks have superior generalization, but if made too simple, unable to learn sparse features needed to reliably discriminate between classes.

Procedure for architecture design included creation of a deep enough structure to reliably identify class variations, as to avoid under-fitting. During initial training on proposed network architectures, regularizers[1] were disabled, as to iprevent ample bias. Regularization methods were added after established architecture, as to avoid extensive variance, which causes over-fitting. The CNN was built and tested using Tensorflow [46]. With several structures being tested, the network architecture yielding best classification results is shown in figure 3.4.2

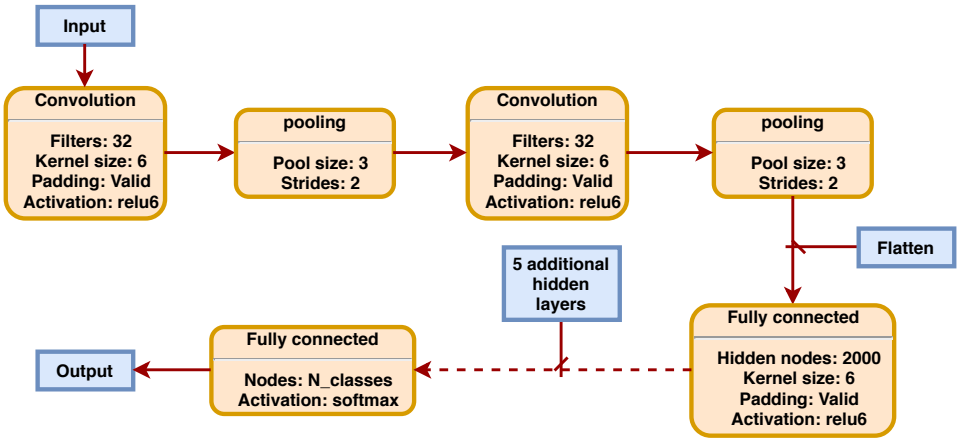

Figure 3.4.2: Convolutional network yielding best results. Two convolutional layers is applied to the volumetric data to extract features from the brain volumes. A 2x2 max-pooling layer for down-sampling follows each convolutional layer. Resulting features are then processed through a series of hidden layers with 2000 hidden nodes, as to learn sparse features concealed in the dataset. The outputs of the the hidden layers are then processed to the output layer, where a probability for each class is calculated, with the classifier picking the class with the highest probability.

The architecture include two convolutional layers, with subsequent 2x2 max-pooling layers. These layers were used for feature extraction and down-sampling

[1] L2-regularization, dropout and early stopping

respectively. 6 fully connected layers follows, addressed to learn resulting feature vectors from previous layers. Learning variations in the data included the amount of hidden nodes being of similar amount as the feature vector size. A softmax affected output layer produces a probability for each of the given classes, with the classifier picking the class of highest probability. The CNN-pipeline is shown in figure 3.4.3

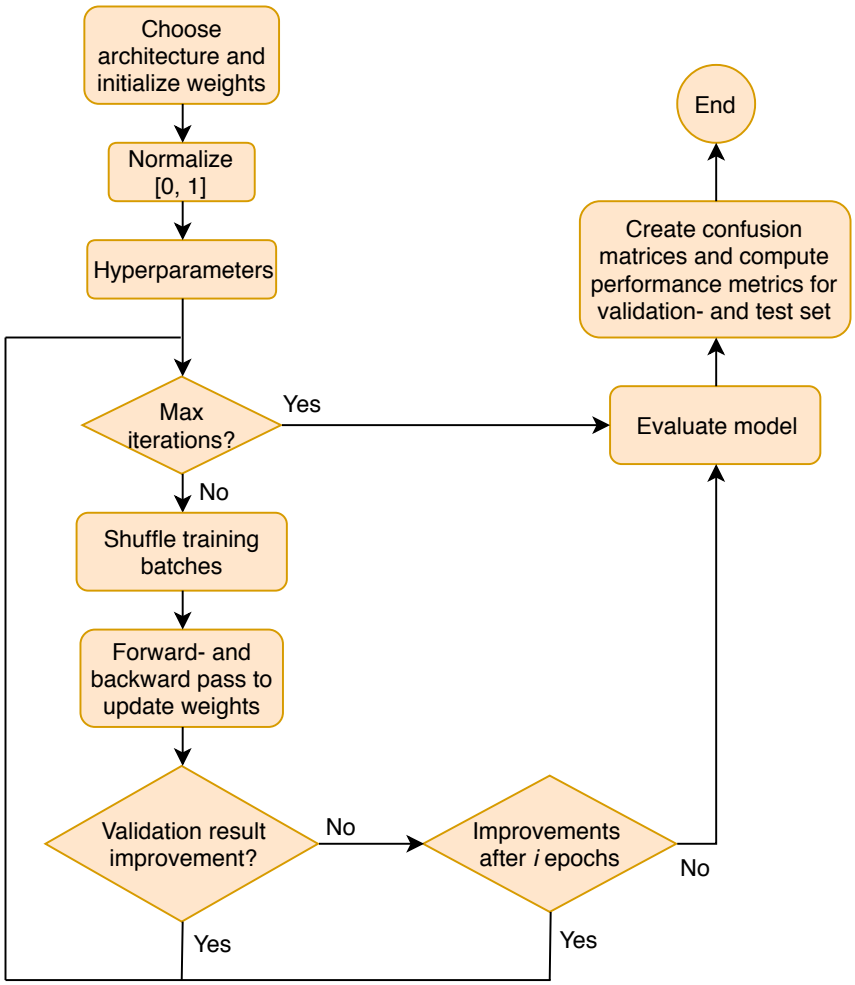

Figure 3.4.3: Architecture is chosen before weights and biases are initialized. The data is min-max normalized [0,1] and hyper parameters values decided. Following initialization, all training data samples are passed through once per one of several epochs, with weights updated based on the error after every finished batch. After every epoch, the network is tested on the validation set. If the validation score does not improve after i epochs, the procedure is terminated, as over-fitting is expected to start. If validation score improves, the network runs until a set number of epochs is finished. For evaluation of network performance, confusion matrices are constructed and performance metrics computed.

Chapter 4

Results

This chapter presents experimental results achieved in this thesis, with testing executed according to methods given in chapter 3.

4.1 Layout

This section presents the parameters used to achieve results for each problem.

Results for both SVM- and CNN-classification were achieved with their respective parameters presented in table 4.1.1.

Problem	CNN					SVM	
	E_p	B_s	η	λ	D_o	C	Kernel
NC/AD/DLB	138	8	0.015	0.02	0.85	0.0072	Linear
AD/DLB	79	5	0.012	0.012	0.82	0.0080	Linear
NC/DLB	80	5	0.01	0.01	0.85	0.0068	Linear
NC/AD	55	5	0.009	0.01	0.85	0.0066	Linear

Table 4.1.1: Hyper parameter values
Abbrevations:
E_p = Total number of epochs
B_s = Batch size
η = Learning rate
λ = L2-factor
D_o = Dropout-factor
C = Error penalizer

4.2 Experimental results

This section presents the experimental results achieved by both the SVM- and CNN-classifier.

Concluding results are presented through the use of performance metrics given in section 2.7.1. Total accuracy visualizes a general probability of predicting correct class, while precision and recall evaluates class-specific measures. Results for the three class problem and each of three binary class problems are presented, with the best overall accuracy highlighted in bold text. Precision and recall are given for each class, as to visualize shortcomings of the classifiers. Results are given in table 4.2.1.

CNN			
Class	TotAcc	Rec_i	Pre_i
NC/AD/DLB	**66.03 %**	0.63	0.52
		0.65	0.62
		0.69	0.96
AD/DLB	74.76 %	0.74	0.75
		0.75	0.74
NC/DLB	**82.14 %**	0.83	0.81
		0.81	0.83
NC/AD	**60.82 %**	0.47	0.67
		0.77	0.59

SVM			
Class	TotAcc	Rec_i	Pre_i
NC/AD/DLB	59.33 %	0.31	0.55
		0.62	0.42
		0.85	0.81
AD/DLB	**80.00 %**	0.81	0.76
		0.79	0.83
NC/DLB	80.00 %	0.83	0.80
		0.77	0.80
NC/AD	58.50 %	0.53	0.61
		0.59	0.51

Table 4.2.1: Results achieved with CNN
Abbrevations:
TotAcc = Total accuracy on test data
Pre_i = Precision for class i
Rec_i = Recall for class i

Chapter 5

Discussion

The chapter discusses achieved results and encountered limitations.

5.1 Classifier performance

The CNN-classifier performed better overall, compared to that of the SVM-classifier. It achieved better results for the three class problem, the NC/DLB- and NC/AD binary problems. Total best accuracy achieved for the three class problem was 66.03 %, while 60.82 %, 80.00 % and 82.14 % were achieved for NC/AD, AD/DLB and NC/DLB respectively. With clinical diagnosis performed to label datasets and co-morbidity a prevalent issue within dementia, classification results are not definite. While subjects with confirmed co-morbidity were judged as outliers and discarded prior to dataset generation, other subjects can suffer non-diagnosed, which affects a generated dataset. With co-morbidity fairly common among patients, flawless classification is improbable and results must be adjudged accordingly.

The CNN-classifier struggles at recognizing NC scans, as given in the recall parameter for the NC/AD problem. Its recall and precision towards NC subjects in the three class problem also falls short compared to other classes. Fairly stable recall rates are found in all problems but the aforementioned NC/AD problem, with strong precision towards DLB scans. Differential diagnosis of AD- and DLB scans obtained balanced recall and precision for either class, with an achieved accuracy of 74.76 %, suggesting that an implemented classifier for differential diagnosis would predict correctly about $^3/_4$ of the time.

The SVM-classifiers also struggles at recognizing NC scans, with recall value in the three class problem particularly worrying. Classification results from differential diagnosis simulation perform marginally better than its CNN counterpart, with class-specific parameters less balanced. The specified results suggest that an SVM-classifier based solely on a single TA-method would predict correctly $^4/_5$ of the time, potentially bettering DL at differential diagnosis. In general, SVM-classifiers are better at recognizing DLB subjects, with recall rates surpassing that of CNN-classifiers. The overall scores of the SVM-classifier does however fall short of its DL counterpart.

Results does not compare favourably with previous studies [2] [47] [48] [49], but enhancement is feasible by atoning for limitations given in section 5.2. The dataset was bigger compared to that of earlier studies with similar problem formulation, with over-fitting on account of dataset size less of an underlying issue in this thesis. With shallower datasets available for comparable studies, cross-validation[1] [50] has often been used, as to utilize the dataset to its full potential. Different cross-validation techniques successfully exploits all data available, but some level of bias affects the resulting classification, with all data used for both training and testing.

5.2 Limitations

This section thoroughly describes limitations encountered during the course of this thesis.

5.2.1 Dataset

The dataset was designed with DLB-data in collaboration with Stavanger University Hospital, which is part of the EDLB-consortium. It was further supplemented with NC- and AD scans obtained from the ADNI-study. AD- and NC data provided from ADNI were restricted to subjects with T1 structural scans available, as most DLB subjects were limited to structural T1 scans. To avoid dataset contamination with regards to similar subjects, first-time scans compromising baseline- and screening scans were selected from the ADNI-study, with follow-up scans of identical subjects duly ignored.

[1] A training set is split into k-parts, with (k-1) working as training data, and the remaining as test. The classifier is trained and tested k times, where each fold works as test set once. Results are presented with a mean and standard deviation over k tests

With the entirity of used AD- and NC scans drawn from the ADNI-study, additional scans of all classes are available from the EDLB-study. A natural progression involves dataset enhancing through addition of available scans, as more high quality data in general equals better performance for deep learning [13]. Inclusion of these subjects can expand the dataset, while preserving balance between classes. Augmentation through T2-Flair inclusion could provide additional features for differential diagnosis [2], but as scans are normally not available for all subjects, a resulting diminishing dataset might prove counterintuitive.

5.2.2 Pre-processing

A conventional pre-processing pipeline includes several methods for minimizing noise factors and possible artefacts. With full-scale brain volumes used directly, both for feature extraction and DL purposes, the entirety of brain volumes contributes to emerging feature space and feature vectors. Deriving optimal regions of interest (ROI) for obtaining features can be explored. Identifying features excellent at highlighting differences in dementia variations and normal controls can be investigated for subsequent boosted classifier performance.

Deriving optimal regions of interest (ROI) for obtaining features excellent at highlighting differences in dementia variations and normal controls, can be investigated for boosted classifier performance.

5.2.3 Texture Analysis and Features

Statistical methods available through the GLCM matrices were used as features for the SVM-classifier. Other methods for texture analysis, including Local Binary Patterns [51] and Histogram of Gradients [52], has yielded promising results in other studies [2].

Previous studies have reported a unique pattern of GM atrophy found within DLB patients that's not present within AD patients [53]. Inclusion of Rapid eye movement sleep behavior disorder (RBD), an early characteristic of DLB that can occur many years before the onset of dementia [54], as a core clinical feature has been reported to improve diagnostic accuracy of autopsy-confirmed DLB [55]. Beta power has also been reported as a feature able to discriminate scans of AD- and DLB patients [49].

It is expected that a larger pool of features, from which the aforementioned have been reported to reliably separate classes, will expose superior traits for discriminating NC-, AD- and DLB controls. The preceding statement is based on feature vector reduction prior to classification, as characteristics that best explains class variation are retained.

Features available were sufficient for separating NC and AD with DLB, but struggled to differentiate NC and AD. This was also, to a lesser degree, the resulting problem for feature extraction within CNN architecture. With all NC- and AD scans obtained through the ADNI-study and all DLB scans obtained through the EDLB-study, subtle similarities might exist between AD- and NC scans that differentiates compared to DLB scans. Even with a similar pre-processing pipeline, elements of these variations might be preserved in resulting brain volumes, assisting the classifier in differentiating DLB better than the other classes, artificially enhancing results.

5.2.4 Architectures

3D-convolutions are immensely time-consuming, and the search for optimal architectures and hyper parameters are problematic within shorter time-frames, including a Master's Thesis. 3D-convolution has been reported to outperform several other classifiers in neuroimaging, including a modest gain over 2D-convolutional models [56]. The pursuit of a flawless model was executed through a series 2D- and 3D-convolutions, with architecture and hyper parameters adjusted towards improved performance. With limited set-ups examined, the structure is not expected to have achieved its optimum. When establishing the eventual architecture, its resulting network was not able to reliably determine all training samples correctly. This implies that the final architecture experienced some bias.

With implementation of a proposed ROI-method as given in section 5.2.2, CNN training can become less overwhelming. Decreased input size prompt both reduced computational power and architecture intricacy, greatly decreasing overall complexity. Diminishing noise factors on account of reduced input size can provide improved details for discriminating sparse features between classes.

Chapter 6

Conclusion

This thesis explores and compares DL- and SVM CAD on T1-weighted MRI of NC-, AD- and DLB patients. Similar pre-processing steps prior to method implementation were executed, with smoothing only being performed prior to deep learning classification, as to retain vital spatial information when performing TA.

Results on a dataset of 760 subjects does not directly suggest if either method outperforms the other. Classification results are relatively similar, with both methods having room for improvement. Previous studies [2] [47] [48] [49] suggests there are better results available, with more textural analysis methods implemented. Previous deep learning studies recommend greater datasets over intricate models based on inferior datasets [57].

The CAD developed in connection with this thesis can perform classification on T1 structural MRIs, although results are fairly inconsistent. Its various implementations are able to handle a three-class problem directly, or any one of the possible binary classifications. Models are dependent on provided test data following a standard pre-processing pipeline before usage, for the given results in chapter 4 representing expected outcome of classification. With results fairly erratic, the classifiers can't be relied upon for predicting correct diagnosis unassisted, but can be performed in addition to a doctor's diagnosis for verification.

It is suggested that results examined during the course of this thesis is built upon and improved, with potential for superior CAD-structures available.

6.1 Future work

To achieve thorough overview of SVM- and DL-classification in neuroimaging, with all their pros and cons, obtaining a larger dataset is desirable. Performing a series of classifications on a dataset, with quantity of included data adjusted for each set, could yield results implying data amount required to experience better results with deep learning compared to that of texture analysis methods.

While an obvious solution for acquiring more data would be including data from other studies, which can involve strenous work, artificial dataset increase through various data augmentation methods could be explored, as data can be used more efficiently [58] and reduce over-fitting [41]. Data augmentation methods include translational- and rotational adjustments to image data, as to artificially create new training samples.

Persisting in search of an optimal architecture that can discriminate classes reliably can aid the classifier, by being able to learn sparse features concealed in volumetric data. Identifying such an architecture can limit the bias of classifiers, abling networks to recognize different properties for particular classes. Deriving a ROI that best describes structural variations between classes could introduce features reliable for differential diagnosis.

Extracting more features that can better express differences in the various classes could help improve overall accuracy, with PCA retaining features that best explains data variations. PCA excels when features are expected to correlate strongly, as a reduced set of uncorrelated features can be generated. With additional TA methods introduced, amount of different layouts producing correlating characteristics can be reduced. Different feature reduction methods can be explored and considered, as other methods can be preferable with reduced correlation. Obtaining additional MR sequences or other modalities such as PET scans for all subjects could aid the classifier in distinguishing classes, as potential superior features could be established before feature vector reduction. Previous studies indicates that there exists more textural information in T1 structural images compared to the FLAIR images [2], but a combination might improve results.

Bibliography

[1] G. M. McKhann, D. S. Knopman, H. Chertkow, B. T. Hyman, C. R. Jack, C. H. Kawas, W. E. Klunk, W. J. Koroshetz, J. J. Manly, R. Mayeux *et al.*, "The diagnosis of dementia due to alzheimer's disease: Recommendations from the national institute on aging-alzheimer's association workgroups on diagnostic guidelines for alzheimer's disease," *Alzheimer's & dementia: the journal of the Alzheimer's Association*, vol. 7, no. 3, pp. 263–269, 2011.

[2] K. Oppedal, T. Eftestol, K. Engan, M. K. Beyer, and D. Aarsland, "Classifying dementia using local binary patterns from different regions in magnetic resonance images," *Journal of Biomedical Imaging*, vol. 2015, p. 5, 2015.

[3] A. Alzheimer's, "2015 alzheimer's disease facts and figures." *Alzheimer's & dementia: the journal of the Alzheimer's Association*, vol. 11, no. 3, p. 332, 2015.

[4] "Dementia statistics," https://www.alz.co.uk/research/statistics, accessed: 2018-02-05.

[5] "Amyloid plaques and neurofibrillary tangles," https://www.brightfocus.org/alzheimers/infographic/amyloid-plaques-and-neurofibrillary-tangles, accessed: 2018-05-31.

[6] S. Sarraf, G. Tofighi *et al.*, "Deepad: Alzheimer's disease classification via deep convolutional neural networks using mri and fmri," *bioRxiv*, p. 070441, 2016.

[7] Z. Walker, K. Possin, B. Boeve, and D. Aarsland, "Lewy body dementias." *Lancet (London, England)*, vol. 386, no. 10004, p. 1683, 2015.

[8] J. Quinn, "Dementia," 2014.

[9] J. J. Zarranz, J. Alegre, J. C. Gómez-Esteban, E. Lezcano, R. Ros, I. Ampuero, L. Vidal, J. Hoenicka, O. Rodriguez, B. Atarés *et al.*, "The new mutation, e46k, of α-synuclein causes parkinson and lewy body dementia," *Annals of neurology*, vol. 55, no. 2, pp. 164–173, 2004.

[10] S. V. Jones and J. O'brien, "The prevalence and incidence of dementia with lewy bodies: a systematic review of population and clinical studies," *Psychological medicine*, vol. 44, no. 4, pp. 673–683, 2014.

[11] G. Litjens, T. Kooi, B. E. Bejnordi, A. A. A. Setio, F. Ciompi, M. Ghafoorian, J. A. van der Laak, B. van Ginneken, and C. I. Sánchez, "A survey on deep learning in medical image analysis," *Medical image analysis*, vol. 42, pp. 60–88, 2017.

[12] S. M. Plis, D. R. Hjelm, R. Salakhutdinov, E. A. Allen, H. J. Bockholt, J. D. Long, H. J. Johnson, J. S. Paulsen, J. A. Turner, and V. D. Calhoun, "Deep learning for neuroimaging: a validation study," *Frontiers in neuroscience*, vol. 8, p. 229, 2014.

[13] J. Cho, K. Lee, E. Shin, G. Choy, and S. Do, "How much data is needed to train a medical image deep learning system to achieve necessary high accuracy?" *arXiv preprint arXiv:1511.06348*, 2015.

[14] S. Klöppel, C. M. Stonnington, J. Barnes, F. Chen, C. Chu, C. D. Good, I. Mader, L. A. Mitchell, A. C. Patel, C. C. Roberts *et al.*, "Accuracy of dementia diagnosis—a direct

comparison between radiologists and a computerized method," *Brain*, vol. 131, no. 11, pp. 2969–2974, 2008.

[15] R. Li, W. Zhang, H.-I. Suk, L. Wang, J. Li, D. Shen, and S. Ji, "Deep learning based imaging data completion for improved brain disease diagnosis," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2014, pp. 305–312.

[16] J. N. Giedd, "Structural magnetic resonance imaging of the adolescent brain," *Annals of the New York Academy of Sciences*, vol. 1021, no. 1, pp. 77–85, 2004.

[17] A. S. of Neuroradiology, "Acr-asnr practice guideline for the performance and interpretation of magnetic resonance imaging (mri) of the brain," 2013.

[18] I. G. McKeith, D. Galasko, K. Kosaka, E. Perry, D. W. Dickson, L. a. Hansen, D. Salmon, J. Lowe, S. Mirra, E. Byrne *et al.*, "Consensus guidelines for the clinical and pathologic diagnosis of dementia with lewy bodies (dlb) report of the consortium on dlb international workshop," *Neurology*, vol. 47, no. 5, pp. 1113–1124, 1996.

[19] I. G. McKeith, B. F. Boeve, D. W. Dickson, G. Halliday, J.-P. Taylor, D. Weintraub, D. Aarsland, J. Galvin, J. Attems, C. G. Ballard *et al.*, "Diagnosis and management of dementia with lewy bodies fourth consensus report of the dlb consortium," *Neurology*, vol. 89, no. 1, pp. 88–100, 2017.

[20] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," in *european conference on computer vision*. Springer, 2014, pp. 346–361.

[21] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE transactions on evolutionary computation*, vol. 1, no. 1, pp. 67–82, 1997.

[22] A. C. Evans, D. L. Collins, S. Mills, E. Brown, R. Kelly, and T. M. Peters, "3d statistical neuroanatomical models from 305 mri volumes," in *Nuclear Science Symposium and Medical Imaging Conference, 1993., 1993 IEEE Conference Record*. IEEE, 1993, pp. 1813–1817.

[23] M. Reuter, N. J. Schmansky, H. D. Rosas, and B. Fischl, "Within-subject template estimation for unbiased longitudinal image analysis," *NeuroImage*, vol. 61, no. 4, pp. 1402–1418, 2012. [Online]. Available: http://dx.doi.org/10.1016/j.neuroimage.2012.02.084

[24] W. D. Penny, K. J. Friston, J. T. Ashburner, S. J. Kiebel, and T. E. Nichols, *Statistical parametric mapping: the analysis of functional brain images*. Elsevier, 2011.

[25] F. Ségonne, A. M. Dale, E. Busa, M. Glessner, D. Salat, H. K. Hahn, and B. Fischl, "A hybrid approach to the skull stripping problem in mri," *Neuroimage*, vol. 22, no. 3, pp. 1060–1075, 2004.

[26] M. Mikl, R. Mareček, P. Hluštík, M. Pavlicová, A. Drastich, P. Chlebus, M. Brázdil, and P. Krupa, "Effects of spatial smoothing on fmri group inferences," *Magnetic resonance imaging*, vol. 26, no. 4, pp. 490–503, 2008.

[27] G. Castellano, L. Bonilha, L. Li, and F. Cendes, "Texture analysis of medical images," *Clinical radiology*, vol. 59, no. 12, pp. 1061–1069, 2004.

[28] D. Chen, S. Li, Z. Kourtzi, and S. Wu, "Behavior-constrained support vector machines for fmri data analysis," *IEEE transactions on neural networks*, vol. 21, no. 10, pp. 1680–1685, 2010.

[29] R. M. Haralick, K. Shanmugam *et al.*, "Textural features for image classification," *IEEE Transactions on systems, man, and cybernetics*, no. 6, pp. 610–621, 1973.

[30] A. Baraldi and F. Parmiggiani, "An investigation of the textural characteristics associated with gray level cooccurrence matrix statistical parameters," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 33, no. 2, pp. 293–304, 1995.

[31] J. Friedman, T. Hastie, and R. Tibshirani, *The elements of statistical learning*. Springer series in statistics New York, 2001, vol. 1.

[32] S. Wold, K. Esbensen, and P. Geladi, "Principal component analysis," *Chemometrics and intelligent laboratory systems*, vol. 2, no. 1-3, pp. 37–52, 1987.

[33] T. Hope, Y. S. Resheff, and I. Lieder, "Learning tensorflow: A guide to building deep learning systems," 2017.

[34] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

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

[36] A. Krizhevsky and G. Hinton, "Convolutional deep belief networks on cifar-10," *Unpublished manuscript*, vol. 40, p. 7, 2010.

[37] M. D. Zeiler, "Adadelta: an adaptive learning rate method," *arXiv preprint arXiv:1212.5701*, 2012.

[38] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[39] S. Geman, E. Bienenstock, and R. Doursat, "Neural networks and the bias/variance dilemma," *Neural computation*, vol. 4, no. 1, pp. 1–58, 1992.

[40] A. Y. Ng, "Feature selection, l 1 vs. l 2 regularization, and rotational invariance," in *Proceedings of the twenty-first international conference on Machine learning*. ACM, 2004, p. 78.

[41] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[42] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, "Scikit-learn: Machine learning in python," *Journal of machine learning research*, vol. 12, no. Oct, pp. 2825–2830, 2011.

[43] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern classification*, 2001.

[44] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[45] L. Al Shalabi, Z. Shaaban, and B. Kasasbeh, "Data mining: A preprocessing engine," *Journal of Computer Science*, vol. 2, no. 9, pp. 735–739, 2006.

[46] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

[47] 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.

[48] P. Vemuri, G. Simon, K. Kantarci, J. L. Whitwell, M. L. Senjem, S. A. Przybelski, J. L. Gunter, K. A. Josephs, D. S. Knopman, B. F. Boeve *et al.*, "Antemortem differential diagnosis of dementia pathology using structural mri: Differential-stand," *Neuroimage*, vol. 55, no. 2, pp. 522–531, 2011.

[49] M. Dauwan, J. J. van der Zande, E. van Dellen, I. E. Sommer, P. Scheltens, A. W. Lemstra, and C. J. Stam, "Random forest to differentiate dementia with lewy bodies from alzheimer's disease," *Alzheimer's & Dementia: Diagnosis, Assessment & Disease Monitoring*, vol. 4, pp. 99–106, 2016.

[50] R. Kohavi *et al.*, "A study of cross-validation and bootstrap for accuracy estimation and model selection," in *Ijcai*, vol. 14, no. 2. Montreal, Canada, 1995, pp. 1137–1145.

[51] 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, no. 7, pp. 971–987, 2002.

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

[53] J. L. Whitwell, S. D. Weigand, M. M. Shiung, B. F. Boeve, T. J. Ferman, G. E. Smith, D. S. Knopman, R. C. Petersen, E. E. Benarroch, K. A. Josephs *et al.*, "Focal atrophy in dementia with lewy bodies on mri: a distinct pattern from alzheimer's disease," *Brain*, vol. 130, no. 3, pp. 708–719, 2007.

[54] R. B. Postuma, J.-F. Gagnon, M. Vendette, and J. Y. Montplaisir, "Idiopathic rem sleep behavior disorder in the transition to degenerative disease," *Movement Disorders*, vol. 24, no. 15, pp. 2225–2232, 2009.

[55] T. J. Ferman, B. F. Boeve, G. E. Smith, S.-C. Lin, M. Silber, O. Pedraza, Z. Wszolek, N. Graff-Radford, R. Uitti, J. Van Gerpen *et al.*, "Inclusion of rbd improves the diagnostic classification of dementia with lewy bodies," *Neurology*, vol. 77, no. 9, pp. 875–882, 2011.

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

[57] A. Halevy, P. Norvig, and F. Pereira, "The unreasonable effectiveness of data," *IEEE Intelligent Systems*, vol. 24, no. 2, pp. 8–12, 2009.

[58] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

[59] "Tools for nifti and analyze image," https://se.mathworks.com/matlabcentral/fileexchange/8797-tools-for-nifti-and-analyze-image, accessed: 2018-02-03.

[60] "cooc3d," https://se.mathworks.com/matlabcentral/fileexchange/19058-cooc3d, accessed: 2018-03-03.

[61] "Glcm texture features," https://se.mathworks.com/matlabcentral/fileexchange/22187-glcm-texture-features, accessed: 2018-03-01.

Appendices

Appendix A

Appendix

This appendix lists the packages required to successfully run scripts and function. Used scripts are briefly described.

A.1 Python

The Python scripts requires the following packages, given in list A.1.

- Tensorflow
- Scikit-learn
- Pandas
- Numpy
- Seaborn
- Scipy
- OS
- Time
- Matplotlib

The Python scripts constructed to perform the necessary tasks is given in list A.1. The dataset is distributed to Python through matfiles containing the 3D-volumes and their corresponding labels.

- **CNN.py**
  Pipeline for performing CNN-classification in Python. Hyper parameters, which classes to test and architectures can be changed by the user.

- **SVM.py**
  Pipeline for SVM-classification. Parameters, PCA and which classes to test can be changed by the user.

- **Functions.py**
  The functions created for and called upon by the aforementioned scripts.

A.2 Matlab

The Matlab packages requires the following packages, given in list A.2.

- SPM12

- Tools for NIFTI and ANALYZE image [59]

The Matlab scripts constructed to perform the necessary tasks is given in list A.2.

- **Main.m**
  The main file used to call upon separate functions to perform required tasks.

- **Preprocessing.m**
  The main file for the pre-processing pipeline.

- **TextureAnalysisFeatExt.m**
  The main file for TA and feature extraction.

- **Normalize.m**
  Batching process for spatial normalization of volumetric data to MNI152 space.

- **Segment.m**
  Batching process for segmenting brain tissues from noise factors.

- **Skullstrip.m**
  Batching process for reconstruction of brain volumes.

- **Smoothing.m**
  Batching process for smoothing the volumetric data with a 5x5x5 kernel.

- **cooc3D.m**
  Creates a gray level co-occurence matrix for 3D data [60].

- **GLCM_Features.m**
  Extracts statistical features from gray level co-occurence matrices[1] [61].

- **xml2struct.m**
  Converts information from xml-files to struct-format. Used for extracting of metadata.

- **csv2data.m**
  Script for acquiring metadata for EDLB subjects[2].

- **Extract_Metadata_ADNI_xml.m**
  Script capable of extracting metadata from the xml-template of ADNI metadata files.

- **Extract_Nifti.m**
  Script digging through directories, extracting all available nifti files.

- **loadniftispm.m**
  Function for loading nifti files for SPM12.

- **Realign.m**
  Function for brain volume realignment. Only used for fMRI, if available

- **load_untouch_nii.m**
  Function for loading nifti files.

A.3 Excel

Excel files used during this thesis is given in list A.3.

- **EDLB_clin_demo_info**
  Metadata available for EDLB subjects

[1] Copyright (c) 2008, Avinash Uppuluri. All rights reserved
[2] Courtesy of Postdoc researcher Ketil Oppedal