UNIVERSITY OF AMSTERDAM
Faculty of Science

BSC KUNSTMATIGE INTELLIGENTIE

BACHELOR'S THESIS (18 EC)

3D group equivariant CNNs for autism classification

Author
Jochem S. Soons
11327030

Supervisor
dr. R.M. Thomas &
Mr D. Arya

Informatics Institute
Faculty of Science
University of Amsterdam
Science Park 904
1098 XH Amsterdam

June 28th, 2019

Abstract

Convolutional neural networks (CNNs) have delivered the most promising results within the recent emergence of deep learning applications in neuroimaging research. In this thesis, a recently developed CNN-based technique known as group equivariant convolutional neural networks (group-CNNs) have been implemented and tested on the ABIDE dataset, which comprises of structural and functional MRI data of patients suffering from Autism Spectrum Disorder (ASD). Moreover, 3D group-convolutions were applied to fully exploit the volumetric MRI-data provided by the ABIDE dataset. To determine whether there were possible benefits to using 3D group-convolutions in classifying on ASD, performance of the 3D group-CNN has been compared with a standard 3D CNN as baseline method. The results demonstrate that the 3D group-CNN outperformed the standard 3D CNN, reporting higher general classification scores in terms of accuracy and AUC. Moreover, results also indicate the ability of group-CNNs to converge faster and achieve better results when trained on less data. The latter effect of reduced sample complexity could prove to be particularly valuable in the neuroimaging context, as data is usually scarce in this domain. Altogether, the results reveal the potential that group-CNNs might have to offer in neuroimaging research, and therefore some recommendations for future research are provided in the final section.

Acknowledgements

First and foremost, I would like to thank my supervisors dr. R.M. Thomas and Mr D. Arya for providing me the opportunity to work on this specific research subject, and for guiding me throughout my research process. They have helped me in setting up the main directions of my thesis, and they were always available for meeting with me to discuss any questions I had regarding the project. Secondly, I would also like to thank Ahmed El-Gazzar, who is a PhD Candidate at the AMC Department of Psychiatry. Ahmed provided me some useful assistance in setting up the technical implementation of my research, by sharing his own repository with me. Lastly, I would also like to thank my co-student Lisa Salomons for her work and helpfulness, as we implemented some technicalities together in early stages of our research, that we would both use in our individual approach that followed in later stages.

Contents

Chapter 1

Introduction

The increasingly large amount of labelled digital information available in nearly every domain of technology has caused major leaps in the development of Artificial Intelligence (AI), and in particular machine learning methods [1]. These AI techniques have taken full advantage of this emergence of 'big data' and have subsequently achieved impressive results in many different domains [2, 3]. One of these affected domains is the health-care sector, that because of its richness in complex data and processes, has exciting prospects for the implementation of modern machine learning techniques [2].

One particular field of interest within the application of AI in the medical domain is that of psychiatric neuroimaging research. The primary goals of this research field are to study the human brain's functional connectivity and to identify objective bio-markers that may inform the diagnosis and treatment of brain-based disorders such as Alzheimer's disease, ADHD, autism and strokes [4]. Moreover, resting state functional magnetic resonance imaging (rs-fMRI) [5] has become the most popular neuroimaging technique in studying the brain's functional connectivity [6–9]. Connectome data that can be obtained via fMRI techniques, offers a rich source of neuroimaging data representing the functional connectivity of the human brain. Therefore, application of machine learning algorithms to this specific brain imaging data possesses great potential [10]. More specifically, applying machine learning methods in classification tasks using fMRI data could increase diagnostic accuracy of various psychiatric and neurological disorders, speed up the diagnostic process, and help in unveiling the hidden pathological brain patterns that serve as bio-markers for these brain-based disorders [11–19].

Various machine learning techniques have been applied in the context of neuroimaging research with different levels of success. Recently, however, a particular family of machine learning methods known as deep learning has conceived considerable attention, because these deep learning techniques have delivered impressive results in a growing number of task areas such as image classification and speech recognition [3]. Moreover, deep learning techniques have become an increasingly popular method of choice in neuroimaging research because they are particularly good in extracting subtle patterns and features from data [3, 10, 20]. Within the recent rise of applying deep learning-based methods in neuroimaging context, recent research has shown that convolutional neural networks (CNNs) have generated the most encouraging results in classifying and predicting psychiatric and neurological disorders [3]. CNNs are a specific class of deep neural networks, that have become a popular and commonly used method in computer vision and image classification tasks.

In this thesis, a recently developed CNN-based method, known as group equivariant CNNs (Group-CNNs or G-CNNs), will be implemented and tested on the Autism Brain Imaging Data

Exchange (ABIDE) dataset. The ABIDE dataset comprises of neuroimaging data for more than two thousand samples, of whom approximately half are controls and half are patients suffering from Autism Spectrum Disorder (ASD) [21]. Group-CNNs were recently introduced by [22], and results of implementations have suggested that this new CNN technique reduces sample complexity, thereby making the network more data efficient [23]. This supposed increase in data efficiency could prove useful, because sample sizes used in the field of deep learning-based neuroimaging research are typically not extremely large as in other domains where deep learning is applied [23–25]. Moreover, group-CNNs have hitherto not been applied to connectome data, allowing it to possibly offer other interesting and insightful perspectives in the field of neuroimaging classification.

To evaluate the performance and possible contribution of group equivariant CNNs on the binary classification task posed by the ABIDE dataset, results will be compared with a standard CNN, that serves as the baseline method. Because the fMRI data in the ABIDE dataset is three-dimensional (i.e. a 3D image of the brain), both the standard CNN and the Group-CNN will be using 3D convolutions to fully explore the volumetric properties of the dataset. The main goal of this study is thus to explore how 3D Group-CNNs and standard 3D CNNs differ in performance when they are tested on the ABIDE dataset. The research question of this thesis can therefore be formulated as the following:

'How do group equivariant 3D-CNN and standard 3D-CNN classification techniques compare in performance when classifying on Autism Spectrum Disorder (ASD) neuroimaging data using the ABIDE dataset?'

More specifically, performance of both CNN models is subdivided in three separate aspects: (1) the general classification performance on the entire dataset, (2) the effect of sample size on classification performance, and (3) the rate of convergence in the training process. To get a full understanding of what this all entails, the thesis will be further subdivided in five chapters. First, relevant previous research and important theoretical concepts are introduced in chapter 2. Here, the precise functionality and theoretical differences between the two CNN techniques are explained too. Secondly, in chapter 3, the dataset and preprocessing steps are discussed, a description of the implemented network architectures is given, and the general approach and methods of performance evaluation are outlined. Thirdly, in chapter 4, the results of the experiments are presented. In chapter 5, a conclusion is provided that highlights the most important findings of this thesis. Finally, shortcomings and recommendations for future research are discussed in chapter 6.

Chapter 2

Theoretical framework

In this chapter, all relevant theoretical concepts of the research in this thesis are discussed. First, the interrelation between ASD as mental disorder and functional MRI as neuroimaging technique will is discussed briefly in section 2.1. Secondly, in section 2.2 the emergence of deep learning in neuroimaging context is reviewed, to offer some perspective on the background of the neuroimaging research field. Thirdly, an explanation about the functionality of convolutional neural networks is provided in section 2.3. Finally, in section 2.4, the concept of group equivariant convolutional networks is explained.

2.1 Autism and functional MRI

Autism spectrum disorders (ASD) emerge in early life and are generally associated with lifelong disability [26]. ASD affects roughly 1% of the population, and is characterized by difficulties in social interactions, communications impairments and a narrow repetitive pattern of interests and behaviours [27,28]. These behavioural manifestations cover social, communicative and cognitive domains, making ASD an intriguing disorder, of which it is difficult to discover a unifying causal explanation [29]. Although it has been established that ASD is a disorder with a genetic and neurological origin, no specific gene or local brain area has yet been designated as the explanation of the psychiatric disease [29]. Given the breadth and depth of symptoms of ASD, in combination with the fact that no ASD explanatory local brain area has been discovered, research has strongly suggested that ASD is causally related to a systems-level neural abnormality [29–32]. This implies that ASD is characterized by alternations in the functional connectivity of the whole brain instead of some local brain abnormality.

A major contributor to the understanding that ASD is related to alternations in functional connectivity of the brain has been the development of functional MRI (fMRI) techniques, that exploit differences in magnetic properties of oxygenated and deoxygenated blood to produce an indirect measure of neuronal activity [29, 33]. Whereas structural MRI techniques provide a detailed map of the brain's tissues and structures by capturing shape, size, and integrity of gray and white matter structures in the brain at one single time point, fMRI detects changes of blood flow within the brain over a time-series, thereby capturing the functional connectivity between remote regions. Moreover, fMRI generates 3D volumetric data representing the brain's functional connectivity, through the sequential acquisition and accumulation of multiple 2D slices. In addition, functional connectivity can be captured when a patient is performing a specific task or when a patient is in a resting or task-negative state (resting-state fMRI) [34]. By offering a wealth of useful information, MRI-based techniques have formed the basis of neuroimaging research in

the past few decades [10].

2.2 Deep learning in neuroimaging research

This section will elaborate on the recently developed interest in deep learning methods within neuroimaging research. To achieve this, developments and shifts in methodology within the neuroimaging research field will be discussed over four stages.

2.2.1 Traditional methods

In the last two decades, MRI-based neuroimaging studies of psychiatric and neurological patients have relied heavily on analytical techniques that typically compare patients with a diagnosis of interest against disease-free individuals, reporting neuroanatomical or neurofunctional differences at group level. Although these univariate analytical techniques have contributed to significant advances in our understanding of the neurobiological basis of psychiatric and neurological disorders, there are two major limitations of these traditional techniques [3]. First, statistical inferences are drawn from multiple independent comparisons of different parts of the brain, based on the assumption that these different local brain regions act independently and functional connectivity can be disregarded. However, just as it was stated earlier that bio-markers for ASD are probably best found by analyzing changes in functionality of the whole human brain rather than local alternations, multiple studies have concluded that this network-level approach should be taken for several other psychiatric and neurological disorders too [35–37]. Secondly, traditional neuroimaging statistical techniques can be used to expose differences between groups, but they do not allow statistical inferences at the individual level. In daily clinician practice, however, diagnostic and treatment decisions need to be made at this individual level [3].

2.2.2 Machine learning

To try to overcome the limitations of traditional methods, a growing interest in machine learning methods has been developed within the neuroimaging community. Machine learning methods are multivariate and can therefore take the inter-correlation between different brain regions into account, thereby overcoming the first posed limitation of the traditional univariate techniques [38]. Furthermore, statistical inferences at individual subject level are possible using machine learning methods, so the second limitation is also overcome [3]. By overcoming the two major limitations of traditional techniques, machine learning applications within neuroimaging research have shown to substantially improve fMRI-based classification for fast and objective diagnosis of mental disorders, such as autism [11, 39], schizophrenia [24, 37], major depressive disorder [40], and cognitive impairment [41]. Most of these machine learning studies applied traditional classification algorithms, such as support vector machines (SVMs) and least absolute shrinkage and selection operator (LASSO). Despite the popularity of these techniques, they have been criticized for not performing well on raw data and requiring medical knowledge that is used to extract the most informative features out of the high-dimensional fMRI data [3, 4, 20]. This step is known as feature selection and leads to the usage of these extracted features as input data for classification, instead of using the raw data [1].

2.2.3 Deep learning

While machine learning methods like SVM remain a popular technique in classifying tasks using neuroimaging data, deep learning recently gained considerable attention in the neuroimaging community [3]. Deep learning methods are a kind of representation learning methods, meaning they can learn relevant features within datasets automatically without the requirement of feature selection [42]. Moreover, because deep learning methods involve hierarchical structures with different levels of complexity, increasing levels of abstraction can be achieved, enabling high-level features to be extracted from data that are less affected by noise in the input data [3]. Following these attributes of deep learning methods, an essential difference between deep learning and traditional machine learning is that in machine learning methods features are usually manually designed to reflect properties of the connectome data seen as important from a medical expert's point of view, while in deep learning methods features are learned from the raw input data, requiring no prior knowledge. Therefore, applying deep learning techniques ensures a more objective and less bias-prone process [3]. Moreover, deep learning methods are better in detecting complex and subtle patterns in fMRI data because they possess the ability to achieve higher levels of abstraction and complexity relative to other machine learning techniques [20].

2.2.4 Overcoming challenges

Taking the strong properties of deep learning previously discussed into account, deep learning applications in neuroimaging classification tasks seem promising and arguably superior to other machine learning methods. However, two big challenges arose when deep learning was firstly introduced to the field of neuroimaging. The first major challenge is that training deep learning methods on fMRI data is computationally expensive. Hence, computational limitations kept deep learning from being applied on a large scale in neuroimaging research [42]. However, stronger graphics processing units (GPUs) have been recently developed that can be used with lower costs, thereby limiting the problem of computational requirements in deep learning research [3, 42]. The second and arguably largest challenge in applying deep learning methods to neuroimaging data is that deep learning-based classification methods perform outstandingly well on classifying tasks when they can be trained on immensely large image databases like ImageNet [43] (which consists of more than 14.000.000 images of over 100.000 different objects), but usually perform significantly worse when training data is scarce [44, 45]. This problem of data scarcity became and remains a challenge for applying deep learning methods in the neuroimaging domain, because here labeled connectome data is typically not available in large quantities [24, 25]. Besides, acquiring additional data is difficult, because collecting fMRI data is an expensive and time-consuming task [42, 46]. Moreover, ethical constraints limit accessibility of privacy-sensitive neuroimaging data for research purposes [2].

Although acquiring large-scale neuroimaging datasets that can compete with dataset sizes in other deep learning domains remains a challenge, international data-sharing initiatives have been developed that aggregate functional and structural brain imaging data collected from laboratories around the world. Examples of such initiatives, varying in sample sizes from a few hundred to a few thousand subjects, are the Autism Brain Imaging Data Exchange (ABIDE) [47], the Alzheimer's Disease Neuroimaging Initiative (ADNI) [48], and the ADHD-200 database [49]. With the recent availability of these publicly-available, relatively large neuroimaging data sets such as ABIDE for training purposes, deep learning-based methods for automatic diagnosis of psychiatric disorders have not only become feasible, but also significantly successful in improv-

ing classification accuracies of mental disorders such as Alzheimer's disease [50–52], autism spectrum disorder (ASD) [4, 53], ADHD [54], schizophrenia [24] and more [3]. Moreover, the most promising results within these growing number of studies that that successfully applied deep learning techniques to neuroimaging data were generated using convolutional neural networks as prediction method [3].

2.3 Convolutional Neural Networks

Convolutional neural networks (CNNs) are a kind of deep neural network whose architecture is loosely based on the biological structure of the visual cortex within the human brain, which is the region of the brain that processes visual information [55]. Essentially, a CNN is an artificial neural network that consists of one input layer, one or multiple hidden layers and one output layer, all of which consist of nodes or neurons that are connected to neurons in subsequent layers (see Figure 2.1) [3]. Following the biological inspiration in design, CNNs were initially introduced to process images, and although CNNs are nowadays used for a variety of tasks, image classification remains its most popular application. When implemented for classification purposes, CNNs fall under the machine learning branch of supervised learning, meaning a CNN is essentially an algorithm capable of generalizing rules or patterns from a labeled set of input data, and using that knowledge to make predictions and classifications on previously unseen data [56].

Although CNNs have been around since the 1990's, the technique experienced a breakthrough in popularity within the field of computer vision after Alex Krizhevsky and his colleagues developed AlexNet in 2012, which is a deep learning-based CNN [44, 57]. In 2012, AlexNet competed in the ImageNet large scale visual recognition challenge and won with a top-5 error rate of 15.3%, which was more than 10.8% lower than the error rate of the runner up. After the success of AlexNet lots of follow-up research has been done in developing different versions and problem-specific types of CNNs, but the main technical components of these CNNs are approximately the same. The generic architecture of a CNN is shown in Figure 2.2. To provide some essential understanding of the functionality of CNNs, seven important concepts will be discussed: (1) convolution, (2) non-linearity (ReLU), (3) pooling, (4) classification, (5) optimization, (6) regularization and (7) model dimensionality. Each of these seven components is discussed in the subsections below, followed by a brief overview of the state-of-the-art autism classification results achieved using a CNN-based approach. Moreover, the general information in following subsections about CNNs mainly stems from the online Stanford course CS231n, which contains a detailed explanation of CNNs applied in computer vision [58].

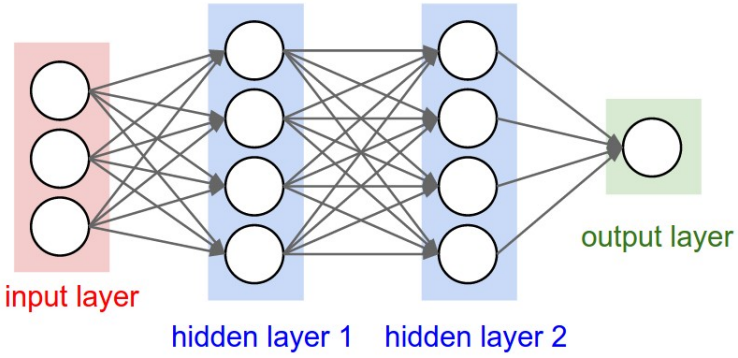


Figure 2.1: Abstract architecture of a neural network [58].

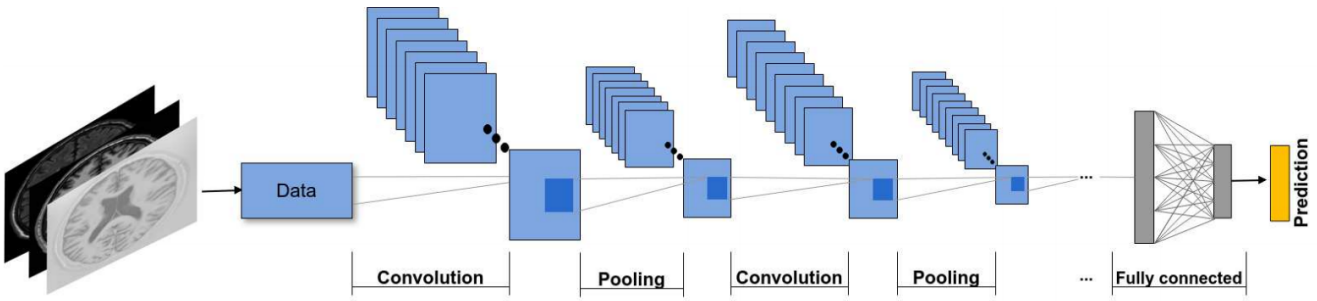


Figure 2.2: Generic architecture of convolutional neural networks [42].

2.3.1 Convolution

The main purpose of convolutional layers in a CNN is to extract features from an input image. The mathematical details of the convolution step will not be discussed here, but instead an intuitive understanding of how convolution works will be presented. A global overview of the convolutional operation is given Figure 2.3. Essentially, three elements are involved in the convolution operation: the input image (or input features in deeper layers), a filter and a feature map.

Firstly, an input image can be represented as a 2D matrix, in which pixel values are represented by numbers. Secondly, the filter, often also referred to as feature detector or kernel, is a smaller matrix (e.g. a 3x3 matrix) that slides over the input image. This smaller matrix, that can be seen as a small patch or window, consistently slides over different positions in the input image and, for every position, computes the element wise matrix multiplication between the filter and the part in the input image that is currently 'covered' by the filter. The output of this computation per position results into a single number, that subsequently forms a single element in the output matrix. This output matrix is referred to as the feature map and is completed when the filter has slid over the entire input image. The feature map, which is a 2D matrix, is now smaller in dimensionality than the input image, but the main idea is that filters detect specific filter-dependent features (e.g. lines or curves) whose presence is represented in the corresponding feature map. So, put differently, a specific filter learns the presence of a certain feature in a input image, and subsequently represents this presence into a resulting feature map. In a convolutional layer, normally multiple filters are applied, that develop a feature map one by one and stack these feature maps against each other. This results in the output of a convolutional layer being a stack of feature maps.

Stride and padding

As was previously mentioned, feature maps are a bit smaller in dimensionality because of the filter that slides over the original input image. Because drastic dimensionality reduction is sometimes desired in convolutional layers (e.g. because of computational efficiency), a technique known as strides can be incorporated in the convolution operation. Essentially, the convolution operation starts at the top-left corner of some input image, and then it slides over all locations both down and to the right. If stride is set to the default value of 1, this sliding window approach is taken by sliding one pixel at a time. However, if stride is set to some higher value (e.g. 2), then the window is moved with two pixels at a time, thereby skipping intermediate locations and decreasing the dimensionality of the output feature maps. On the other hand, it may also

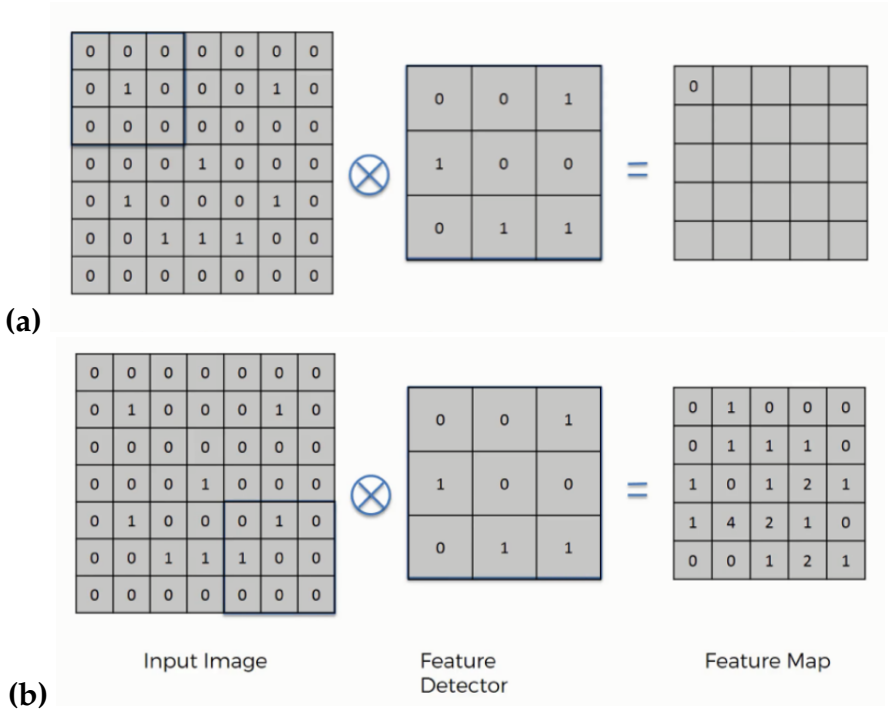


Figure 2.3: Convolution operation [59]. **(a)** displays the start of the convolution operation (the first value of the feature map being calculated), while **(b)** displays the end of the convolution operation (the last value being calculated).

occur that dimensionality reduction is not desirable. In this case, a technique known as padding can be applied. Padding adds extra zero-valued pixels around the boundary of the input image, thus increasing the effective size of the input image. In practical implementations, padding can either be set to 'valid', meaning no padding is applied, or to 'same', which means that a border of zero-valued pixels is added that would keep output feature maps the same dimension as in the input image (if stride would be set to default value 1).

2.3.2 Non-linearity (ReLU)

ReLU stands for Rectified Linear Unit and is a non-linear activation function, that is applied after each convolutional operation. The main purpose of ReLU is to introduce non-linearity in the CNN. ReLU is applied to introduce this non-linearity, because all convolutional operations are linear operations (e.g. matrix multiplications), and it is beneficial for neural networks to learn well on non-linear data too. The output of the ReLU activation of a node is given by Equation 2.1. Following from Equation 2.1, ReLU essentially replaces all negative pixel values in the feature map with zero. This leads to a property of CNNs called sparsity, which means that certain neurons in the CNN are not activated at all in producing the eventual output because their activation is set to zero. This is often a desirable property of CNNs, because these 'dead' neurons inside networks can lead to concise models that have better predictive power and less overfitting (which will be discussed later). Moreover, in sparse networks it is more likely that activated neurons are processing meaningful features of the input image. In addition, computational processes in a sparse network also become faster than in a dense network, as there are fewer things to compute. The downside of scarcity in networks, however, is that too many neurons may die, which can cause deterioration in performance and a large part of the network to be regarded as

useless. Solutions for this problem may be the implementation of a "leaky" ReLU activation function, that prevents neurons from dying. Besides the possible benefit of sparsity, ReLU also has an advantage over other activation functions because it has low computation costs, as there is no complicated math. This makes ReLU currently the most popular and used activation function in CNNs.

$$\mathrm{R(x)} = max(0, x) \qquad \textbf{(ReLU activation)} \tag{2.1}$$

2.3.3 Pooling

After the feature maps that have been formed in the convolutional layer are rectified by the ReLU function, usually another type of layer is passed through in the CNN: the pooling layer. The main function of pooling layers in CNNs is ensuring dimensionality reduction of the feature maps while retaining the most important information in those feature maps. This dimensionality reduction is useful, because it makes input representations smaller en therefore more manageable. Furthermore, the number of parameters and computations is strongly reduced in the network, thereby controlling computational requirements. This makes the pooling operation a necessary and important functionality within the architecture of CNNs.

Spatial pooling or sub-sampling can be of different types (e.g. max, average or sum pooling), but the underlying process is very similar. In the process of max pooling, similarly to convolutional layers, a local spatial neighbourhood is defined iteratively via a sliding window approach. This spatial neighbourhood or region is defined through a small window (e.g. a 2x2 region) and iteratively, for every position that the window can take, the largest occurring element in the local region of the feature map is taken en put into the output matrix of the pooling layer. The schematic overview of this operation is shown in Figure 2.4. Instead of taking the largest element, the average of the elements could also be taken as output element (average pooling), or the sum of all elements (sum pooling), or some other function that calculates a value over the local region in the feature map. However, max pooling has proved to significantly outperform other sub-sampling methods in image processing tasks and is therefore in practice probably the best choice of pooling function [60].

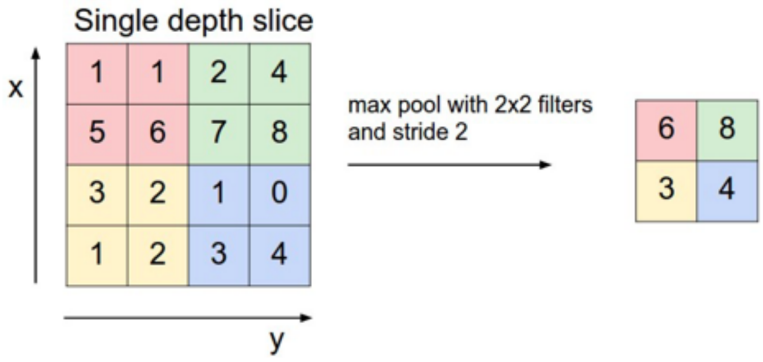


Figure 2.4: Max Pool operation [58].

2.3.4 Classification

In order to get actual class predictions as output from a CNN, in the final part of the model one or more fully connected layers are passed through. Here, 'fully connected' implies that unlike in convolutional layers (where the filters can be considered as the nodes), every neuron in fully connected layers is connected to every neuron in the following layer (see Figure 2.7a). While convolutional and pooling layers serve to represent high-level features in the input image, fully connected layers are usually a convenient method of learning non-linear combinations of these features. This mining of meaningful features from an input image is used for the classification task that is done in the final fully connected layer. The activation of the SoftMax function in the final layer produces prediction probabilities per class that sum up to one. These prediction probabilities are subsequently used for classifying the initial input image as one of the target classes.

2.3.5 Training and optimization

In short, when imaging data is processed by a CNN, images are given as input to the network and when all the previously described layers in the network have been passed prediction scores are returned for those images (the feed-forward pass). CNNs are usually build of multiple layers that can consist of hundreds of nodes that are itself connected to nodes in adjacent layers, resulting in thousands or millions of connections between nodes within a network. Within the feed-forward process of generating output predictions from input images, each node within a layer produces output that is subsequently being passed on as input for the connected node in the following layer, similar to the transmission of electrical signals by neurons in the human brain. The output of an individual node is produced by multiplying the input value by some initial weight value θ and, optionally, adding a bias term β is to this outcome (see Equation 2.2). Consequently, viewed from a CNN as a whole, output is determined by the individual weight values of all nodes in the feed-forward process. To allow a network to make useful predictions, all these weight values (the parameters) are adjusted in a process known as training.

In the training process of a CNN, a method known as backpropagation is used, which is schematically shown in Figure 2.5 [1]. In the training process of a CNN, images from the training set are passed through the network and subsequently, the network's output prediction based on its current configuration of parameters is quantitatively compared with the true prediction of the images (e.g. the target labels of the images). This comparison is done through an error function, also referred to as 'loss function'. In the backpropagation process, starting from the output layer moving towards the previous layers, the weight values of all nodes in the network are adjusted slightly in a direction to minimize the network's loss function (the backward pass). This slight adjustment is determined by multiplying the derivative of the loss function (i.e. the gradient of the loss function) by a user-selected learning rate η, which determines how large the learning steps are (see Equation 2.3). The technique of minimizing the loss function is known as gradient descent, and is schematically displayed in Figure 2.6.

Updating weight values is done in an iterative manner, processing the training images over multiple epochs, with one epoch being one full cycle of processing all the images in the training set. Furthermore, the update of the weight values can be undertaken after computing the loss function for a single training sample (stochastic gradient descent), a small subset of the training set (mini-batch gradient descent), or the whole training set (batch gradient descent). Updating values using the whole training set as batch can be computationally expensive and, hence,

mini-batch gradient descent is most commonly used in the research field, leading to smoother parameter updating and more stable convergence [42]. The optimization process is continued until convergence of the loss function has occurred, which means that the network has reached a state in which the configuration of parameters is optimal. When updating the weight values no longer leads to a decrease of the loss function, the model has converged and can be considered as trained.

$$\text{Output}(\mathbf{x}) = \theta\mathbf{x} + \beta \qquad \textbf{(node output)} \tag{2.2}$$

$$\theta^{updated} = \theta - \eta \cdot \frac{\delta}{\delta\theta}\mathcal{L}(\theta) \qquad \textbf{(weight update)} \tag{2.3}$$

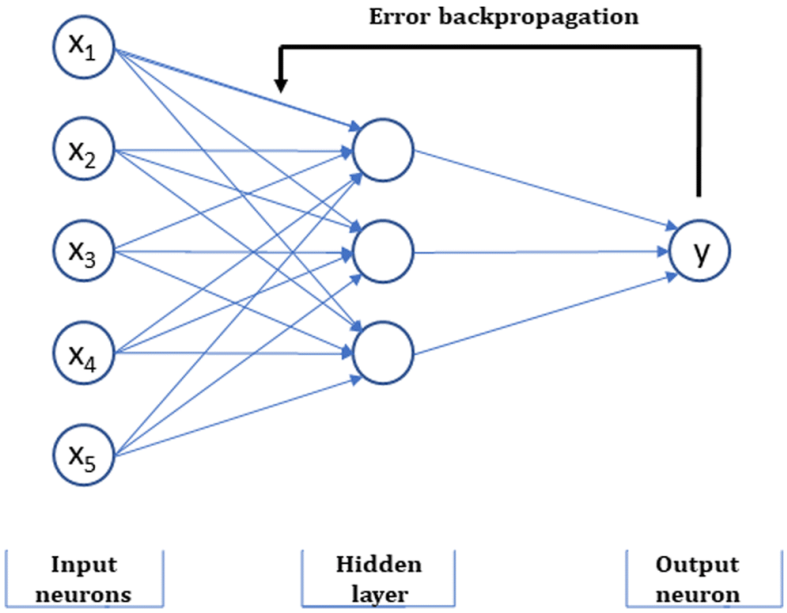


Figure 2.5: Schematic representation of the backpropagation process [2].

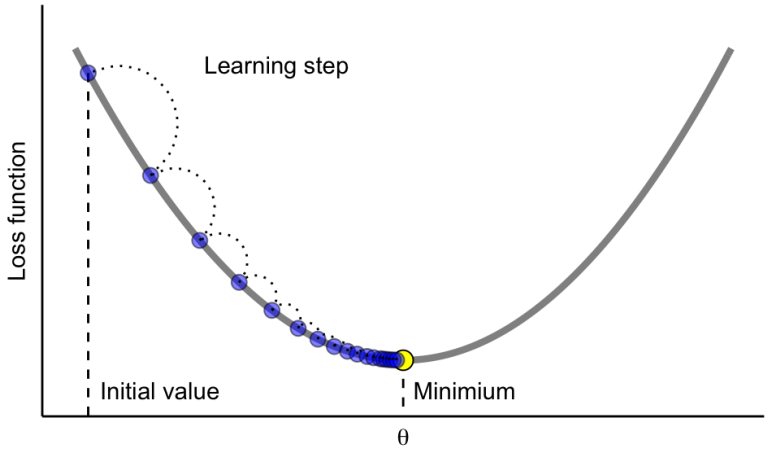
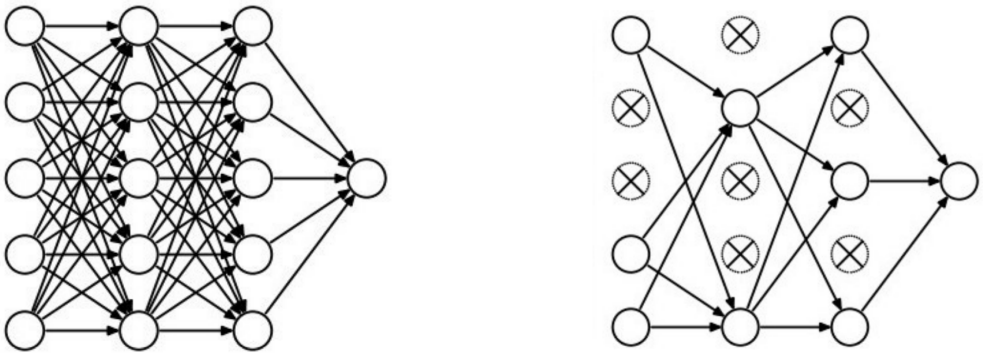


Figure 2.6: Gradient descent method to minimize loss function [61].

2.3.6 Overfitting and regularization

As explained in the previous section, a CNN learns by adjusting an extremely large number of parameters so that the loss on the training set is minimized. After training, the desirable property of a CNN is that it can also make good predictions on previously unseen data (i.e. a test dataset). This property of high predictive power is also known as generalizability. However, because training CNNs involves estimating a large number of parameters, it is possible that the model learns particular fluctuations in the training data that are irrelevant for the purpose of classification on test data - an issue known as "overfitting" [3]. When a CNN overfits, the model will perform very well on the training set but will perform poorly on out-of-sample test data, thereby lacking the desired characteristic of generalizability [62]. The risk of overfitting is particularly high when deep learning methods such as CNNs are applied to neuroimaging data, because the sample size of fMRI-based datasets is usually relatively small, while the dimensionality of the data is very high [3, 11, 63]. In other words, neuroimaging datasets usually consist of few samples, which in turn contain a lot of information and features, thereby increasing the likelihood of overfitting to the training data [64].

However, several strategies have been developed that can be used the mitigate the risk of overfitting, collectively known as "regularization". In addition to the previously discussed regularization contribution of the ReLU activation function (by causing network sparsity), two other popular regularization strategies will be discussed here. The first strategy, known as dropout, is a computationally inexpensive technique that randomly selects neurons to be ignored during the training process. This causes network sparsity in a manner similar to the sparsity caused by ReLU (see Figure 2.7) [62, 65]. Dropout causes neurons to be randomly "dropped-out", which means that their contribution to the activation of neurons in the forward pass is temporally removed and that any weight updates in the backpropagation process are not applied to these neurons. The effect of dropout is that a network becomes less sensitive to specific weights of neurons, which results in a network that can achieve higher levels of generalizability and is less likely to overfit on the training data [62]. The second commonly used regularization strategy is batch normalization, which is a technique that that normalizes the output of a previous layer by subtracting the batch mean and dividing by the batch standard deviation. By doing this, each batch is preprocessed to achieve zero mean and standard deviation equal to one before entering the next layer [66]. Batch normalization has proven to improve both speed and generalizability in training neural networks and has therefore become a popular instrument to boost the performance and stability of CNNs [67].



(a) Fully connected neural network

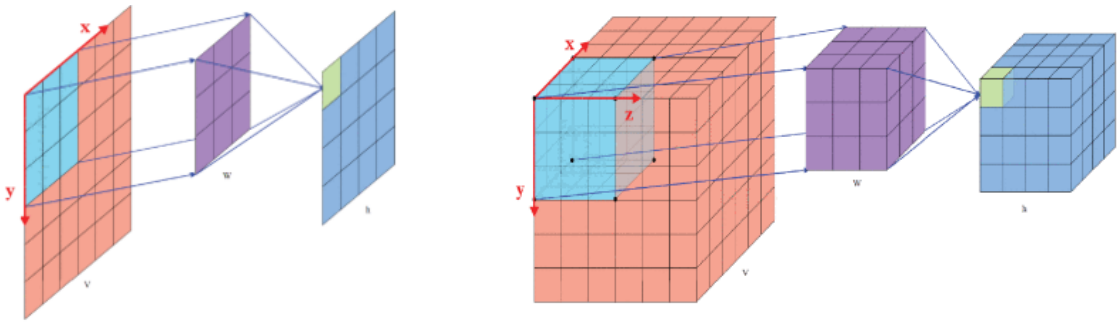
(b) Applying dropout with $p \approx 0.5$

Figure 2.7: Dropout applied to fully connected layers.

2.3.7 Model dimensionality

So far, a general understanding of how traditional CNNs work has been provided. Traditional image processing CNNs have 2D convolutional filters that can be used to process 2D data. However, because MRI data is three-dimensional and 2D convolutional filters can only capture 2-dimensional spatial information, it is challenging to apply traditional CNNs to this 3D connectome data. Applying 2D CNNs to neuroimaging data would mean that input data is flattened in a certain way, resulting in the negligence of information along the third dimension, which can be an important source of predictive information [42, 53].

To address the issue of 2D CNN applications to 3D data, [68] extended the idea of a regular 2D convolution to a 3D convolution operation that could be used to capture both space and time information for video classifications. This approach resulted in the creation of a 3D CNN, that could effectively incorporate motion information of 3D video data. Similar to video classification applications, 3D CNNs can also be implemented to neuroimaging data to fully preserve and exploit the 3D spatial information [53]. Essentially, 3D CNNs can effectively exploit this volumetric data by using 3D kernels (e.g. a 3x3x3 matrix) to slide over 3D input images in the convolutional and pooling layers, instead of using traditional 2D kernels that slide over 2D images (see Figure 2.8). Moreover, research has shown that using 3D convolutions on MRI data usually leads to better performance than using 2D kernels [42]. The main constraint of a 3D CNN approach, however, lies in its expensive computational costs and memory requirements [42].



(a) 2D convolution **(b)** 3D convolution

Figure 2.8: The convolutional process in different dimensions.

2.3.8 CNN-based autism classification

As was previously mentioned, recent applications of CNNs have achieved the most promising results in neuroimaging context [3]. Considering the classification of ASD, the highest accuracy of classifying all participants in the ABIDE database so far was obtained by [53]. This research implemented an ensemble learning strategy using 3D CNNs, which means that a separate 3D CNN models were trained on a variety of parcellations of the brain into smaller regions of interest. Subsequently, separate prediction scores corresponding to these parcellations were pooled together before performing the final classification. Using this CNN-based technique, a classification accuracy of 73.3% was achieved, which significantly exceeding the state-of-the-art previously set by another deep learning-based research [4], in which an accuracy score of 70% was achieved on the ABIDE dataset.

2.4 Group equivariant convolutional neural networks

Now that the general concept of CNNs has been discussed, this section will explain the functionality of group equivariant CNNs (group-CNNs). First, the strong attribute of translational invariance in standard CNNs is explained, by discussing the concept of equivariance in convolutional layers. The second subsection presents works and techniques that aim to extend the notion of equivariance in CNNs, including the recent development of group-CNNs. The third subsection provides a more detailed explanation of how group-CNNs work when they are applied to 3D data. In the last subsection, the promise of applying group-CNNs on neuroimaging data is emphasized.

2.4.1 Translational invariance

In abstract mathematical sense, invariance is defined as the property of some object to remain unchanged when transformations are applied to that object [69]. In machine learning context, invariance with respect to some transformation refers to the ability of a model to produce the same output, whether or not some transformation has been applied to the input (see Equation 2.4). Convolutional neural networks are approximately translation invariant. This is a very strong attribute of CNNs, because this means that spatial translations within an image generally do not influence the ability of a trained model to correctly extract features out of an image. Put differently, the property of translation invariance allows trained CNNs to detect objects or features, even if they are shifted around in images [10, 23].

Whereas a CNN as a whole contains the property of translation invariance, individual convolutional layers within the network are translational equivariant [22]. In the convolutional step, the same filter is applied to every input position through a sliding window approach, causing the weight value of a node in the convolutional layer not to be linked to a specific input position. Instead of learning and optimizing a weight value for every input position, in a convolutional layer only one set of weight values is learned (the filters) and these weights are subsequently shared across input positions. This principle is called weight sharing. By using the same weight (filter) to analyze different positions in an image, CNNs use far less parameters than fully connected models, while retaining the ability to successfully learn features from input data [22]. Moreover, by virtue of weight sharing, if a network learns to recognize a certain feature in one part of the image, then it will be able to do so in any other part as well [10,70]. Additionally, weight sharing ensures that a shift in the input of a convolutional layer results in the same output, but shifted. This concept, in which a transformation (e.g. a translation) applied to input leads to the same transformation applied to the output is known as equivariance (see Equation 2.5) [70]. The attribute of translational equivariance in convolutional layers means that shifting an image and then feeding it through a number of layers has the same effect as feeding the original image through the same layers and then shifting the resulting feature map [22]. This enables CNNs to exploit translations not just in the first, but also in the deeper layers of the network [22, 23].

The translational invariance of CNNs is a direct result of the translational equivariance of convolutional layers combined with dimensionality reduction and removal of spatial information that takes place in the pooling layers [23]. The property of translational invariance contributes strongly to the great successes achieved by CNNs in the computer vision domain, as features and objects can be translational shifted around in input images without causing a strong reduction in performance (see Figure 2.9a) [10]. However, CNNs are naturally not invariant to several other symmetries that are present in perception tasks, such as rotations or reflections.

While humans can relatively easily recognize objects even when they are rotated or reflected, a CNN typically has difficulties with classifying these transformed objects correctly (see Figure 2.9b) [71]. Intuitively, because the translational equivariance in the convolution operation forms such a strong attribute of CNNs, it is reasonable to expect that exploring equivariance of other transformations like rotations in convolutional layers would improve performance even further.

$$f(\mathbf{x}) = f(T_g(\mathbf{x})) \qquad \textbf{(Invariance)} \tag{2.4}$$

$$f(T_g(\mathbf{x})) = T_g(f(\mathbf{x})) \qquad \textbf{(Equivariance)} \tag{2.5}$$

(a) Translational shifts applied to the same object in an image.

(b) Rotations and reflections applied to the same object in an image.

Figure 2.9: Six images with label "dog". A trained CNN would typically have no difficulties in classifying the three images in **(a)**, due to the attribute of translational invariance that CNNs possess. However, a typical CNN would have difficulties in classifying the images in **(b)** because CNNs are normally not invariant to rotations and reflections.

2.4.2 Extending the concept of equivariance

The intuitive notion that invariance and equivariance to rotations would add to performance of CNNs was supported by empirical research done by [72], who showed that the hugely successful AlexNet CNN [44] trained on the large-scale ImageNet database spontaneously learns representations that are equivariant to flips, scaling and rotation. One way CNNs could achieve some sort of invariance to rotations is by implementing data augmentation [73]. This technique involves generating additional training data that consists of original training images on which rotations have been performed. By adding these artificially generated rotated images to the training set, a CNN becomes more robust to rotations in test images, because it is trained on transformed images that it might encounter. However, the disadvantage of this approach is that CNNs only become approximately invariant with respect to the training data. Besides, creating artificial

data is not a very elegant solution in combatting invariance issues in CNNs. Instead of using data augmentation techniques to achieve invariance to some transformation, it is naturally more desirable to build invariance and equivariance into a network itself.

Following this idea, [74] introduced four operations (slice, pool, stack and roll) that could be inserted into an existing CNN framework to build convolutional layers that are partially rotation equivariant, particularly for rotations of 90 degrees. The approach of [74] is based on rotating feature maps after the convolution operation, although they acknowledge that the rotation of filters would be an equivalent method. An approach in which feature maps are copied and rotated instead of filters has the advantage of a relatively uncomplicated implementation, but the main disadvantage of this approach is increased memory requirements. Applying rotational transformations to filters instead of feature maps would reduce memory requirements, as filters are usually much smaller in dimensionality than feature maps, and it would therefore require less memory to copy and rotate these filters [23].

This filter transformation-based approach forms the core of the research by [22], in which the general concept of group equivariant CNNs (Group-CNNs/G-CNNs) is introduced. Whereas the focus of [74] lies in the efficient implementation of CNNs that are invariant to 90 degree rotations, [22] provide a more general framework based on symmetry groups, which can be used to design CNNs that are invariant to several types of transformations, including rotations and reflections. Group-CNNs achieve this high level of invariance through the use of G-convolutions, which is a new type of layer that contains a substantially higher degree of weight sharing than regular convolutional layers. Essentially, in G-convolutions, besides the traditional translation that is applied to filters (by sliding it over input images), additional transformations like rotations and reflections are applied to the filters too. More specifically, starting with a traditional canonical filter with learnable parameters, several transformed copies are produced, which are then slid (translationally) over the input feature maps to produce a set of output feature maps.

In its initial introduction by [22], group-CNNs proved to substantially increase the expressive capacity of a network without necessarily increasing the number of parameters. This was demonstrated by achieving state-of-the-art results on both the rotated MNIST and CIFAR-10 datasets with negligible computational overhead. Furthermore, because [22] provided a general framework of group-CNNs that could be easily extended to suit other convolutional purposes, [23] has extended the framework to work with 3D G-convolutions that are applicable to volumetric imaging data.

2.4.3 3D group equivariance

In the previous section it was explained that G-convolutions involve creating multiple copies of filters that have undergone some transformation such as a rotation or reflection. However, it was not specified how these transformations on the filters are chosen to result in extended equivariance. The work of [22] has shown that convolutional layers become equivariant to some set of transformations that are applied to the filters in the layer, if these transformations are chosen to form some set of transformations called a symmetry group. In general, the symmetry group of an object is the set of transformations that map that object back onto itself without changing it [23]. For example, a square can be rotated by 0, 90, 180 and 270 degrees without changing the square itself. Therefore, these 90° rotations form a symmetry group for a 2D square. So, if we link this back to the fact that equivariance in a G-convolutional layer arises when the filter-applied transformations form a symmetry group, we can conclude that by applying rotations of 0, 90, 180 and 270 degrees to a 2D filter in a 2D G-convolutional layer, equivariance of (90°) rotations

is achieved.

When we approach the idea of symmetry groups in 3D, however, things get a little more complicated. Essentially, 3D filters in convolutional layers can either be cubicles or rectangular cuboids (see Figure 2.8b). Considering only rotations and reflections as relevant transformations in this thesis, [23] identified four symmetry groups of interest in applying G-convolutional to 3D filters: the group exploring only rotational transformations and the group exploring both rotations and reflections for a cube (O and O_h, respectively), as well as the group exploring only rotational transformations and the group exploring both rotations and reflections for a rectangular cuboid (D_4 and D_{4h}, respectively). Because of simplicity and implementation-related purposes, group convolutions in this thesis will only explore the symmetry group O (the group that explores rotational transformations of cubic filters). Group O is schematically displayed in Figure 2.10. As can be seen from the figure, group O consists of 24 different unique rotations performed on a cubic filter. Therefore, when group-CNNs exploring symmetry group O are applied to the three-dimensional MRI data, each filter in the G-convolutional layers will be copied and rotated uniquely 24 times, before being translated over the input data. The implementation of group-CNNs that transform filters according to the symmetry group O has been made publicly available by [23][1] and will therefore form the basis for the technical implementation of 3D Group-CNNs this thesis.

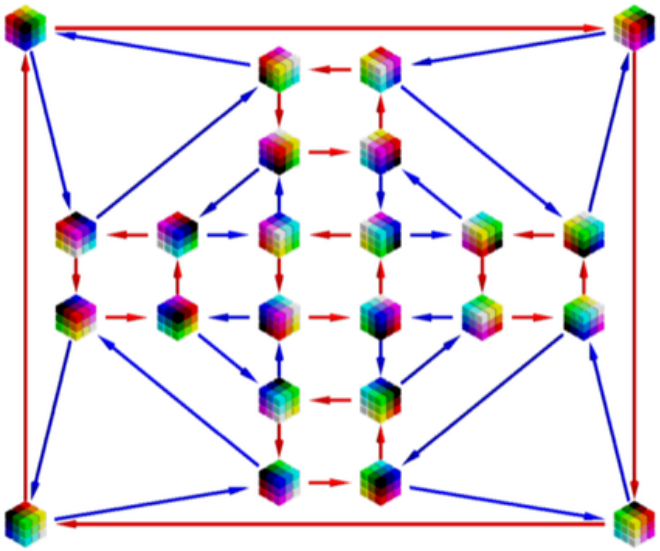


Figure 2.10: Symmetry group O [23]. Red arrows correspond to Z-axis rotation, whereas blue arrows correspond to rotation around a diagonal axis (best viewed in color).

2.4.4 The promise of group-CNNs

After successfully developing 3D group-CNNs, [23] subsequently applied the network to the problem of false positive reduction for lung nodule detection, using 3D CT-scans as input data. The 3D Group-CNNs unambiguously outperformed the baseline standard 3D CNN. Moreover, Group-CNNs even proved to be about ten times more data efficient than the conventional CNNs,

[1]Code can be found on: `https://github.com/tscohen/GrouPy`

meaning they performed about as well as regular CNNs being trained on ten times more data. This apparent attribute of Group-CNNs to reduce sample complexity is especially interesting for domains where data scarcity is an issue, such as in medical imaging [23]. The great promise in the recent results achieved using group-CNNs by [22, 23], in combination with the theoretical advantages of exploring rotational equivariance and the fact that group convolutions have not been applied to neuroimaging data before, makes the implementation of 3D group-CNNs in neuroimaging research a promising one.

Chapter 3

Methodology

This chapter describes the methods that were used to determine whether 3D group-convolutional networks perform better in classifying autism than their regular 3D convolution counterparts. The description of methodology is divided into four sections. In section 3.1, the used dataset and preprocessing steps are discussed. Secondly, in section 3.2 the implementation of both CNN networks is discussed. Thirdly, the global approach of the research in terms of experiments is presented in section 3.3. Lastly, section 3.4 discusses the evaluation metrics that have been used to assess the classification performance.

3.1 Dataset

The ABIDE dataset, which is used as source of data in this research, has aggregated both structural (T1 weighted) and functional (resting-state fMRI) brain images from more than 24 laboratories around the world [21, 47]. The dataset used in this thesis consists of data from both the ABIDE I and ABIDE II phases, resulting in the usage of neuroimaging data from a total amount of 2122 participants, comprising 1007 individuals with a diagnosis of ASD and 1115 typical controls. The dataset includes both males and females (although roughly 80% of the subjects are males) and the total age span of all participants is between 5 and 64 years of age (with median age 13 years). This heterogeneity of samples makes the ABIDE dataset particularly challenging for classification tasks [53]. Before both the structural and functional MRI could be used as input data, multiple preprocessing steps have been taken, which will be discussed in the subsections below.

3.1.1 Preprocessing: structural MRI

As was mentioned in section 2.1, structural MRI imaging provides information about the shape, size, and integrity of gray and white matter structures in the brain. This information can be captured using different structural neuroimaging techniques. In this thesis, the structural MRI acquired from the ABIDE dataset was measured using T1-weighted pulse sequences, which is a technique that provides good contrast between gray matter and white matter tissues, while less contrast is provided between cerebrospinal fluid (CSF) and brain matter [33]. Moreover, the T1-weighted MRI images were preprocessed using FMRIB's Automated Segmentation Tool (FAST) [75], which segments a 3D image of the brain into different tissue types, whilst also estimating and correcting spatial bias fields (also known as spatial intensity variations or radiation field

inhomogeneities). Subsequently, skull stripping was performed using the Brain Extraction Tool (BET) [76], which is a preprocessing method that removes non-cerebral tissues like skull, scalp and dura mater (the membrane that surrounds the brain) from the imaging data.

3.1.2 Preprocessing: resting-state fMRI

As was described in section 2.1, fMRI techniques capture functional connectivity of the brain by detecting blood flows over a time-series. Moreover, 3D images of the brain's functional connectivity are captured through the sequential acquisition of multiple 2D slices. This results in the raw data of a fMRI scan being a time-series of 3D brain images. To make the raw data more suitable as input for the CNNs, several steps have been taken in the preprocessing process, using the Configurable Pipeline for the Analysis of Connectomes (CPAC) [77]. Firstly, effects of subject motion during fMRI scans were reduced using standard motion correction. Secondly, slice timing correction was applied to correct for subtle differences in time acquisition of each 2D slice, so that the signal from the same time point was used in acquiring the whole brain volume. Thirdly, spatial smoothing was implemented to average intensities of neighbouring voxels (which are the equivalent of pixels in 2-dimensional space, i.e. a data point in 3-dimensional space). Spatial smoothing has the effect of blurring images slightly and making them smoother by softening the hard edges, thereby improving the signal-to-noise ratio. Lastly, high-pass filtering was applied to cut off frequencies within the raw signal that were noise-related or not of interest.

After these preprocessing steps were taken, the fMRI data still consisted of 3D volumetric brain images over a time series, making it 4-dimensional. As was described in section 2.3.6, one of the main problems of neuroimaging data is its high level of dimensionality, which increases the risk of overfitting. So, in order to apply the CNNs to the high-dimensional data, it is beneficial to reduce this dimensionality. Since the data concerns functional MRI in which the fourth dimension is caused by time duration of the scan, it is possible to summarize the entire time-series of voxel values into single values, thereby reducing the data to three dimensions [77]. Moreover, summarizing fMRI data can be done using different techniques that compute singular voxel values in their own specific way. In this thesis, each fMRI scan was summarized using 12 different computational techniques. The meaning and underlying measurement methodology of all summaries will be briefly discussed in the following section.

3.1.3 FMRI summaries

Amplitude of Low Frequency Fluctuations (ALFF/fALLF)

ALFF measures the amplitude of low frequency fluctuation of the fMRI signal. Because fluctuations in activity are a fundamental feature of the resting brain's functional connectivity, and the relative magnitude of these fluctuations can differ between brain regions and subjects, ALFF can act as a marker for brain dysfunction [78]. Moreover, fractional ALFF (fALFF) is directly derived from the ALFF measurement. In fALFF measurements, the amplitude of low frequency fluctuations is divided by the whole range of fluctuations, and thus the relative contribution of specific fluctuations to the whole frequency range is represented [79].

Autocorrelation

Autocorrelation measures the statistical dependency of voxel values over a time-series. By measuring dependency of voxel values between contiguous time points, correlation between past

and present states is computed. This way, autocorrelation can act as a marker for functional connectivity of remote brain voxels or regions [80].

Degree centrality (binarized/weighted)

Degree centrality is a measure of local network connectivity that represents the number of direct connections a node (voxel) has with other nodes (voxels) [81]. As such, a brain region with a high number of connections will have a high degree of centrality. Among the many nodes that form the whole-brain network, degree centrality indicates which of these nodes may be considered central within the network. Moreover, in computing degree centrality, binarized degree centrality represents connection strengths between nodes as either 0 or 1 when calculating the number of connections, while weighted degree centrality uses connection strength as a continuous correlation value [82].

Local functional connectivity density (lFCD) (binarized/weighted)

LFCD is another measure of local network connectivity. This technique can be applied to fMRI data to detect nodes (regions) in the human brain with a number of connections that greatly exceeds the average. Essentially, lFCD computes local network connectivity through exploring some node's neighbours and neighbour's neighbours until connections become weaker than some user-given threshold value [83]. Subsequently, if a node has a high number of neighbouring nodes that were explored, a high lFCD value will be assigned, which subsequently indicates high functional connectivity. Similar to degree centrality, binarized lFCD uses connection strength that is represented as 0 or 1 in computation, while weighted lFCD uses connection strengths as continuous values [82].

Eigenvector centrality (binarized/weighted)

Whereas degree centrality is a measure of local network connectivity, eigenvector centrality is a measure of global network connectivity. The eigenvector centrality of a given node in a network is measured by the number of direct connections it has to other nodes that have high centrality. Therefore, the eigenvector centrality of a given node is not only dependent on its own centrality, but also on the centrality of the nodes that it is connected to [81]. As such, a brain region (voxel) with a high eigenvector centrality has connections to many other regions that are themselves highly connected and central within the network. Again, binarized eigenvector centrality uses discrete values for connection strengths in computation, while weighted eigenvector centrality uses continuous values [82].

Entropy

Entropy is a concept that measures complexity within a system [84]. Increased complexity means higher entropy, while reduced complexity means lower entropy. Moreover, brain entropy has been defined as the number of neural states a brain can access [85]. Regions of the human brain are known to organize transiently into functionally connected networks for brief periods only to become reorganized moments later as elements of a network with different structure of functionality [85]. Thus, entropy for a voxel can be calculated from a fMRI time-series by determining the number of (approximately) stable brain states a network has accessed for a brief period of time.

Regional Homogeneity (ReHo)

ReHo provides information about the local activity of regions throughout the brain by evaluating the similarity or synchronization between the time series of a given voxel and its neighbouring voxels [82, 86]. This voxel-based measure of brain activity is based on the hypothesis that brain activity is manifested by clustering groups of voxels, rather than single voxels [82].

Voxel-Mirrored Homotopic Connectivity (VMHC)

VMHC is a measurement of functional homotopy, which is the synchrony in patterns of activity between homotopic (geometrically corresponding) regions in each hemisphere of the brain [87]. Moreover, VMHC measures voxel-wise connectivity between both hemispheres of the brain, by computing the connectivity for all voxels in one hemisphere and their mirrored counterpart in the other hemisphere. Because the variation of functional homotopy can vary between regions and subjects, VMHC can act as marker for brain-based disorders [87].

3.1.4 Preprocessed data: overview

After both the structural and functional MRI were preprocessed, the raw input data was ultimately split into 13 separate datasets: 1 resulting from the T1-weighted structural MRI data, and 12 resulting from summarizing the fMRI data using different techniques. All these separate datasets, which will be referred to as summaries, consisted of 2122 subjects where each sample is represented as a three-dimensional image of dimension 45x54x45. An overview of all summaries with their respective abbreviation and brief specification is given in Table 3.1.

3.2 Network implementation

In this section, implementation details will be provided of both the standard 3D CNN and the 3D group equivariant CNN. This will be done by presenting the architectural design of both networks, and discussing the choice of loss function, optimizer and regularization techniques.

3.2.1 Architectures

Standard 3D CNN

The architecture that was used as standard 3D CNN is shown in Figure 3.1. As can be seen from the figure, the CNN consists of seven layers. 3D Max-pooling was applied in the first and fourth layer, while 3D-convolutions were applied in the second, third and fifth layer using 32, 64 and 128 filters, respectively. Furthermore, both max-pooling operations used kernels of size 2x2x2 with stride set to 2, and all convolutional operations used kernels of size 3x3x3 with stride set to 2 and padding set to 'same' (see section 2.3.1). Moreover, all convolutional layers used ReLU as activation function. After the input had passed through all convolutional and pooling layers, the resulting feature maps were flattened into a 1D-vector before entering the first fully connected layer, containing 256 nodes. In the second fully connected layer, probability scores for the two classes were generated using the SoftMax activation function.

Summary	Abbreviation	Measurement
T1-weighted	$T1_w$	Structural MRI information
Amplitude of Low Frequency Fluctuations	ALFF	Fluctuations of functional activity
fractional Amplitude of Low Frequency Fluctuations	fALFF	Relative fluctuations of functional activity
Autocorrelation	Autocorr	Correlation between brain states
Local Functional Connectivity Density (binarized)	$lFCD_b$	Local network connectivity
Local Functional Connectivity Density (weighted)	$lFCD_w$	Local network connectivity
Degree Centrality (binarized)	DC_b	Local network connectivity
Degree Centrality (weighted)	DC_w	Local network connectivity
Eigenvector Centrality (binarized)	EC_b	Global network connectivity
Eigenvector Centrality (weighted)	EC_w	Global network connectivity
Entropy	Entropy	Network complexity
Regional Homogeneity	ReHo	Regional network activity
Voxel-Mirrored Homotopic Connectivity	VMHC	Functional homotopy

Table 3.1: Overview of all summary datasets.

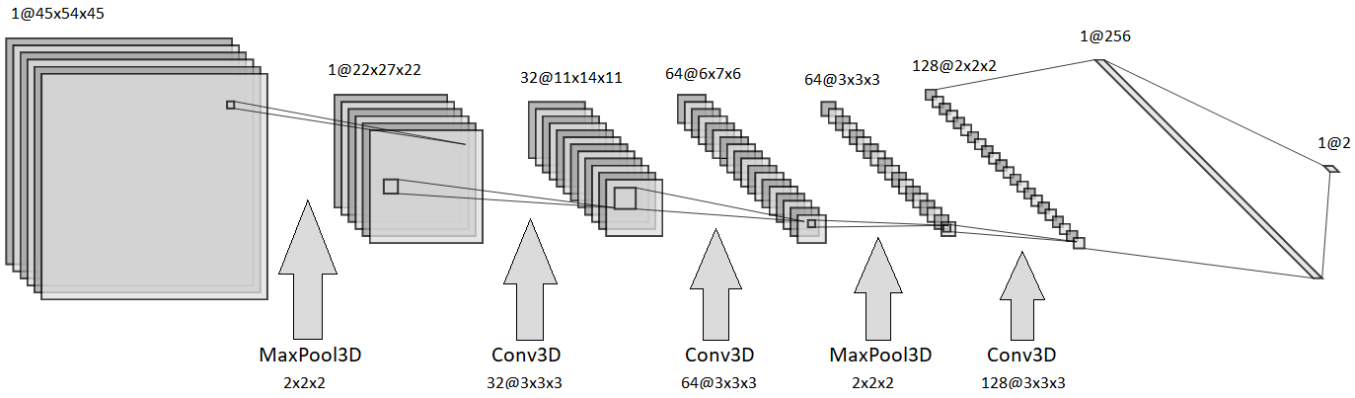


Figure 3.1: Architecture of the standard 3D CNN. The numbers at the top represent the number of feature maps and dimensionality of the data in each layer, while the information at the bottom represent the performed operation with the number of filters and corresponding dimensions.

Group equivariant 3D CNN

Because the main goal of this thesis is to compare performances of the group-CNN and the standard CNN, the implementation of the group-CNN was in terms of architecture as similar as possible to the standard CNN. Moreover, the only actual difference in architecture between the two models was that is that the group-CNN used 3D G-convolutions in the convolutional layers, instead of the regular 3D-convolutions that were used in the standard CNN. However, because G-convolutions exploring symmetry group O copy and transform filters 24 times (see section 2.4.3), using the same number of filters in both the G-convolutions and the standard convolutions would mean that the group-CNN uses significantly more filters than the regular CNN. Therefore, to keep the number of filters and parameters in both CNN models roughly the same, the group-CNN used 6, 10 and 16 filters in each of the three G-convolutional layers, respectively.

3.2.2 Loss function

As was described in section 2.3.5, optimization in a CNN is achieved by minimizing the loss function, which represents the classification error. Different loss functions can be used in image classification tasks, but in this thesis the cross-entropy loss function has been implemented for both CNN models. Cross-entropy, also referred to as log loss, is a popular loss function for binary classification tasks [88], and is mathematically defined as:

$$\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N} -\Big(y_i \cdot \log\big(P(y_i)\big) + (1-y_i)\cdot \log\big(1-P(y_i)\big)\Big) \qquad \textbf{(cross-entropy loss)} \tag{3.1}$$

Where y_i indicates the true label of some sample i (0 or 1) and $P(y_i)$ indicates the predicted probability score of that sample belonging to class 1. Following this equation, and the fact that y can either be 0 or 1, two different cases can be distinguished:

Case 1: A sample's true label is 1 ($y^{true} = 1$), resulting in the loss function:

$$\mathcal{L}(y^{true} = 1) = -\log\Big(P(y=1)\Big)$$

Case 2: A sample's true label is 0 ($y^{true} = 0$), resulting in the loss function:

$$\mathcal{L}(y^{true} = 0) = -\log\Big(1 - P(y=1)\Big)$$

By substituting $\Big(1 - P(y=1)\Big)$ with $P(y=0)$[1], we get:

$$\mathcal{L}(y^{true} = 0) = -\log\Big(P(y=0)\Big)$$

So, what the cross-entropy loss function essentially does is for each sample with true class 1, it adds the negative logarithmic prediction probability of it belonging to class 1, and for each sample with true class 0, it adds the negative log prediction probability of it belonging to class

[1]This follows from the property of the SoftMax function for binary classification, where $P(y=0) + P(y=1) = 1$

0. In the final step of the cross-entropy loss function, the total loss is divided by the number of classified samples, and the remaining mean cross-entropy loss value is returned. By taking the negative log over prediction probabilities, cross-entropy returns high values for wrong classifications with high confidence, while returning low values for correct classifications with high confidence. The plot shown in Figure 3.2 illustrates this: as the predicted probability of the true class gets closer to zero, the loss increases exponentially. This is a desirable characteristic of loss functions in classifying tasks, because it ensures punishment of bad predictions. For this reason, cross-entropy has been chosen as loss function in this thesis.

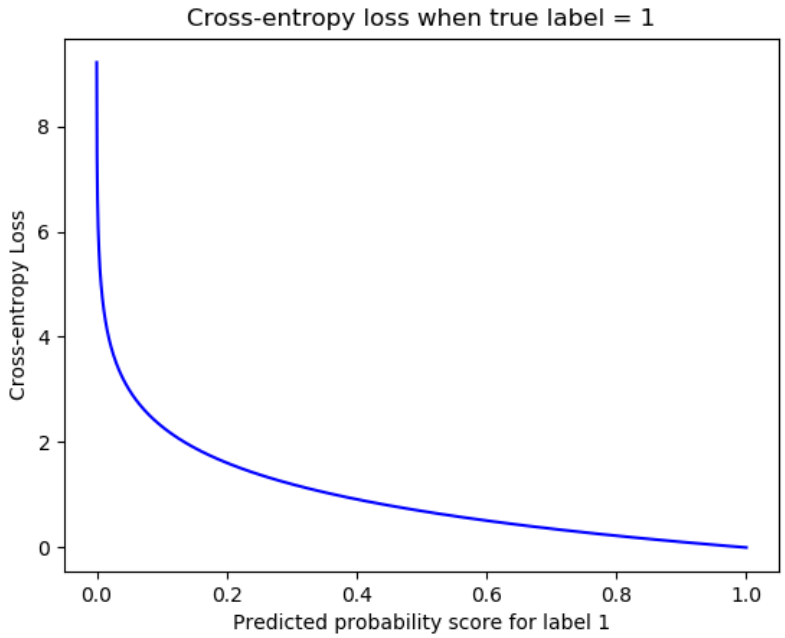


Figure 3.2: Cross-entropy loss function.

3.2.3 Optimizer

In section 2.3.5, it was explained that during training the loss function is minimized using gradient descent methods. In this thesis, the Adam optimizer has been used to minimize the cross-entropy loss, which is an algorithm that can replace traditional stochastic gradient descent methods [89]. Adam is an adaptive learning rate method, which means that it computes individual learning rates for different parameters. Moreover, Adam has shown to improve performance on non-convex optimization problems [42,89]. Non-convex optimization problems are problems that may have several locally optimal solutions, which makes it difficult to identify whether the problem has some global optimal solution [90]. There are several other attractive benefits to using Adam on non-convex optimization problems: the method is straightforward to implement, is computationally efficient, has little memory requirements, and is well suited for problems that are large in terms of data or parameters [89]. These attractive benefits and the fact that multiple empirical results demonstrated that Adam compares favorably to other stochastic optimization methods [89,91], have led to the choice of using the Adam algorithm as optimizer in the training process of the CNNs.

3.2.4 Regularization

As was pointed out before in section 2.3.6, CNNs are particularly susceptible for overfitting when they are trained on a relatively small dataset consisting of high-dimensional MRI data. To reduce the risk of overfitting, batch normalization has been applied in each layer, and dropout has been applied in all convolutional layers, with a dropout probability of 0.5.

3.3 Approach

This section will elaborate on the general approach that has been taken in this research, which ultimately served to compare performance of the standard 3D CNN and the 3D Group-CNN network architectures. Four steps can be distinguished in the global approach, which will be covered in the subsections below.

3.3.1 Data split

The first step was to split all the 13 summary datasets into separate training, validation and test sets. The split of all 2122 samples into these three subsets proceeded with the following proportions: 20% of the data formed the test set, and 80% of all samples formed the basis for the training and validation set, where the training/validation ratio within this 80% of all samples was also 80/20, respectively. An overview of the size of the resulting train, validation and test subsets is displayed in Table 3.2. So, after splitting, each summary dataset is divided into three subsets, thereby creating 13 separate training, validation and test sets.

subset	number of samples
train	1538
validation	340
test	424

Table 3.2: Number of samples in subsets after splitting the dataset.

3.3.2 Hyperparameter selection

Before results of both the standard 3D CNN and the 3D Group-CNN could be compared, it was necessary to first establish hyperparameters that would be used in the training process of the CNNs. These hyperparameters concerned the choice of batch size and learning rate. The hyperparameters were determined by training both networks on the training set of the VMHC summary with different batch sizes and learning rates[2]. Subsequently, it was empirically determined which combination of batch size and learning rate yielded the highest validation accuracy per model and would therefore be used in the rest of the experiments.

[2]VMHC was chosen as summary to train on in determining the hyperparameter values, because in early research stages training on this summary delivered some promising results at the time without overfitting too much. Because it would take significantly more time to extensively test validation performance of all hyperparameter values on all summaries, the choice was made to select the hyperparameter values based only on performance on the VMHC summary.

3.3.3 Summary performance

After the optimal values for both batch size and learning rate were empirically selected, the next step was to evaluate performance of both models on all 13 summary datasets. To do this, both networks were trained on all the 13 summary training sets one by one and tested on corresponding test set. In the training phase, besides tracking performance of the model on the training data, performance on the validation set was also tracked throughout the training process. While training loss may continue to decline in training, a rise in validation loss generally indicates the start of overfitting. Therefore, by keeping track of the validation loss in the training process, a method known as validation-based early stopping can be implemented, which means that the training process is stopped when validation loss no longer decreases. The idea is that this validation-based early stop leads to a higher level of generalizability, and thus better performance on unseen data. However, validation error may fluctuate during training because of local minima, and therefore validation-based early stopping could result in sub-optimal solutions. To prevent this from happening, both CNNs were trained for a fixed number of epochs while keeping track of the validation loss throughout the training process, and a checkpoint of the model was created if the validation loss of the model was lower than the previously lowest validation loss. After training, the configuration of parameters corresponding to the point at which the validation loss was lowest were used for evaluating on the test set. So, instead of implementing validation-based early stopping, a kind of validation-based model extraction technique was implemented, which ensured that the model used for testing achieved lowest validation loss during the training process.

3.3.4 5-fold cross-validation

The performance of the summaries as explained in the previous section was only tested once (using the initial data split), because it would take significantly long to evaluate performance on all 13 summaries using multiple splits of the data in subsets. However, one-time test results can be reasonably biased and misleading, and therefore a somewhat more in-depth performance analysis of both CNNs was the next step in comparing performance. To do this, first the two best-performing summaries were empirically selected from the results of the one-time test results. Subsequently, a 5-fold cross-validation scheme was implemented to generate some more nuanced test results. This means that both summary datasets were split into five folds of equal size, and that during five test iterations, one of these folds acted as the hold-out test set, while the other folds combined formed the training set. Moreover, in these five iterations, the same validation-based model extraction technique as explained earlier was applied by splitting the four non-testing folds into a training set and validation set with ratio 80/20, respectively. In the cross-validation approach taken, multiple evaluation metrics were implemented to measure performance, which will be discussed in detail in the following section.

3.4 Evaluation metrics

As was stated in the introduction, the main goal of this thesis is to compare performance of the standard 3D CNN and 3D Group-CNN, by examining three different aspect: (1) general classification performance based on the whole dataset, (2) the impact of sample size on classification performance, and (3) rate of training convergence. The first subsection below discusses the metrics that have been used to determine general classification performance, by clarifying the terms

accuracy, sensitivity and specificity. The second subsection introduces the concept of ROC-AUC analysis, which is a method that was also used for measuring general classification performance. The third subsection presents the concept of learning curves, which has been the evaluation metric for analyzing the impact of sample size on performance. Finally, the last subsection discusses how rate of convergence has been measured and compared.

3.4.1 Accuracy, sensitivity and specificity

To provide an understanding of concepts of accuracy, sensitivity and specificity, first it is necessary to explain the meaning of the terms true positive, false positive, true negative and false negative. In general, positive means that a subject is identified as patient, while negative means that a subject is identified as healthy. Therefore, the following definitions apply:

- True positive (TP) = correctly identified as patient
- False positive (FP) = incorrectly identified as patient
- True negative (TN) = correctly rejected as healthy
- False negative (FN) = incorrectly rejected as healthy

Accuracy is calculated by dividing the number of correctly classified samples by the total number of samples in the evaluation set, as shown in equation 3.2. Sensitivity, also referred to as the true positive rate (TPR) or recall, measures the proportion of actual positives that are correctly identified as such, and is shown in equation 3.3 [92]. In the context of ASD classification, sensitivity indicates the percentage of ASD patients who are correctly identified as having autism. Specificity, also referred to as the true negative rate (TNR), measures the proportion of actual negatives that are correctly identified as such, and is shown in equation 3.4. In the context of ASD classification, specificity indicates the percentage of healthy people who are correctly identified as being healthy [92]. Sensitivity and specificity are statistical measures that are widely used in medicine, where high sensitivity indicates a medical test that rarely overlooks actual positives (thus ensuring diagnosis if a subject has some disease), and high specificity indicates a medical test that rarely diagnoses a healthy subject as patient (thus ensuring rejection if a subject does not have some disease). Although the optimal medical classifier would have high sensitivity and specificity, in practice there usually exists some sort of trade-off between these two values [93].

$$\text{accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}} \tag{3.2}$$

$$\text{sensitivity} = \text{TPR} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{3.3}$$

$$\text{specificity} = \text{TNR} = \frac{\text{TN}}{\text{TN} + \text{FP}} \tag{3.4}$$

3.4.2 ROC-AUC

Because diagnostic power of any medical test is determined by both its sensitivity and its specificity [93], it is informative to explore the trade-off between these two values in analyzing a classifier's performance. Receiver operating characteristic (ROC) analysis is a method for examining the trade-off between specificity and sensitivity [94, 95]. A ROC curve is a graphical plot that illustrates the diagnostic ability of a binary classifier. The ROC curve is created by plotting the true positive rate (TPR), against the false positive rate (FPR) at various threshold settings (the class discrimination value). The true positive rate is equivalent to the sensitivity (see equation 3.3), while the false-positive rate is also known as the fall-out or probability of false alarm [95]. Furthermore, the false positive rate can be calculated by:

$$\text{FPR} = (1 - \text{specificity}) \tag{3.5}$$

By varying the threshold value, which is the probability score that must be exceeded to classify a subject as being a patient (with a default value of 0.5 in binary classification tasks), different TPR and FPR values are calculated and plotted to form a curve. Because TPR is equivalent to sensitivity and FPR is directly derived from the specificity (see equation 3.5), each point on a ROC curve represents a sensitivity and specificity pair corresponding to a particular decision threshold. Generally, the area under a ROC curve indicates the capability of a model to distinguishing between two classes [94,95]. ROC curves with a high AUC (area under curve) score are better in making predictions than ROC curves with low AUC scores. Moreover, the diagonal in a ROC curve represents random classification, with a corresponding AUC score of 0.5. Classifiers with an AUC score higher than 0.5 are therefore performing better than random classifiers, while classifiers with an AUC score lower than 0.5 are performing worse. Because ROC-AUC analysis offers some useful insights in the diagnostic ability of models, AUC scores were also used to evaluate classification performance in the 5-fold cross-validation scheme, in addition to the previously described accuracy, sensitivity and specificity metrics.

3.4.3 Learning curves

A learning curve measures the impact of the number of training samples on the performance of a predictive model and is constructed by plotting the number of used training samples against some evaluation metric (e.g. accuracy score) that is performed over a constant test set [39]. Two different learning curves have been constructed to compare performance of both CNN models with respect to sample size. The first learning curve plots test accuracy scores against the number of training samples used, while the second training curve plots AUC scores against the number of samples used. In these plots, accuracy and AUC scores were calculated by incrementing the ratio of the used part of the training set each time with 5%, starting with 5% of the training set being used and ending with the entire training set being used. Normally, an increasing learning curve indicates the capability of a model to gain additional useful information when it is trained on more data [39]. In this thesis, however, the usage of learning curves does not necessarily serve to show how much better a model performs on more data, but rather to demonstrate how good a model can still perform when it is trained on less data. In this way, learning curves are used to assess group-CNN's alleged effect of reducing sample complexity, which was discussed earlier in section 2.4.4.

3.4.4 Rate of convergence

As described in section 2.3.5, in the training process of a CNN, the optimizer minimizes the loss function until convergence occurs and overfitting starts to happen, meaning the validation loss does not decrease any longer. To compare the speed of convergence for both CNN models, graphs have been constructed that illustrate the validation loss plotted against the number of epochs passed in the training process. By creating these plots for both the standard 3D CNN and the 3D Group-CNN, it was possible to determine empirically in which model convergence of loss occurred earlier.

Chapter 4

Results

This chapter provides all the results of the experiments that have been conducted to compare the performance of the standard 3D CNN and the 3D group-CNN. An overview of the setup for these experiments can be found in Appendix A. The chapter can be subdivided into five sections. Firstly, in section 4.1, results concerning the selection of hyperparameters are discussed. Secondly, in section 4.2, test results of all summary datasets are presented. Thirdly, results of the cross-validation scheme are discussed in section 4.3. Fourthly, section 4.4 presents the results regarding the analysis of sample complexity. Lastly, in section 4.5, rate of convergence of both models is analyzed.

4.1 Hyperparameter selection

Validation accuracy scores of both CNN models using different combinations of hyperparameters are shown in Table 4.1. In generating these scores, both models were trained for 100 epochs, and validation accuracy was calculated every 5 epochs. After 100 epochs of training were completed, the highest validation score was subtracted and subsequently used as input value for Table 4.1. As can be inferred from Table 4.1, the standard 3D CNN achieved highest validation accuracy (63.1%) with a learning rate of 0.001 and a batch size of 64, while the 3D group-CNN achieved highest validation accuracy (62.7%) with a learning rate of 0.0001 and batch size of 32. However, because both models seemed to be performing relatively well with a learning rate/batch size combination of 0.0001/32, respectively, and because using a batch size of 64 would require more memory capacity in training the models, the decision was made to use a learning rate of 0.0001 in combination with mini-batch size 32 in all further training processes.

4.2 Summary performance

Table 4.2 displays the one-time test results both models for all summaries. Here, both models were trained on all 13 summary training sets for 100 epochs, while using the corresponding validation set to determine which configuration of weight values should be used for evaluating on the hold-out test set. Following from Table 4.2, the 3D Group-CNN slightly outperformed the standard 3D CNN with an average accuracy score of 59.0% for the group-CNN, against 58.1% for the standard CNN. Moreover, both models did not achieve high accuracy scores when they were trained on the structural MRI $T1_w$ summary and the entropy summary, while the top performances of the 3D CNN (62.1% acc.) and 3D group-CNN (62.6% acc.) were achieved

using the ReHo and VMHC summaries as datasets, respectively. Therefore, these summaries were used in the cross-validation process to generate somewhat more nuanced and extensive performance results.

	standard 3D CNN				**3D group-CNN**			
learning rate	**batch size**				**batch size**			
	8	16	32	64	8	16	32	64
0.01	0.583	0.599	0.612	0.589	0.584	0.607	0.578	**0.610**
0.001	0.604	0.604	0.607	**<u>0.631</u>**	**0.600**	0.604	0.618	0.607
0.0001	**0.610**	**0.607**	**0.622**	0.607	0.565	0.596	**<u>0.627</u>**	0.591
0.00001	0.578	0.578	0.589	0.565	0.578	**0.613**	0.594	0.565

Table 4.1: Validation accuracy scores of both CNN models using different hyperparameters. Highest scores per batch size (**bold**) and overall highest score per model (**<u>underlined</u>**) are highlighted.

Summary	**3D CNN**	**3D G-CNN**
$T1_w$	0.553	0.549
ALFF	0.587	0.587
fALFF	0.580	0.592
Autocorr	0.587	0.583
DC_b	0.583	0.618
DC_w	0.587	0.583
$lFCD_b$	0.565	0.571
$lFCD_w$	0.557	0.576
EC_b	0.583	0.594
EC_w	0.587	0.606
Entropy	0.548	0.559
ReHo	**<u>0.621</u>**	0.620
VMHC	0.611	**<u>0.626</u>**
Average acc.	**0.581**	**0.590**

Table 4.2: One-time test accuracy scores of training both CNN models on all summaries. Top performing summaries are **<u>underlined</u>**.

4.3 Cross-validation results

Average results of the 5-fold cross-validation scheme, in which both models were trained for 100 epochs before being tested on each fold, are shown in Table 4.3. It can be noted that the averaged test accuracy scores over the five folds are somewhat lower than the top-performing scores displayed in Table 4.2, implying that the top-performing scores were small outliers. Furthermore, the results in Table 4.3 also show that the 3D group-CNN exceeded accuracy performance of

the standard 3D CNN again. More specifically, the group-CNNs outperformed standard CNNs with accuracy scores of 61.2/61.4% for the ReHo and VMHC summary, against accuracy scores of 60.1/59.7% produced by the standard CNN for both summaries, respectively. Moreover, the group-CNNs also performed better with respect to the sensitivity-specificity trade-off, reporting average AUC scores of 64.2/65.1 for the ReHo and VMHC summaries, against the AUC scores of 63.2/63.6 obtained by the standard CNN. The ROC curves corresponding to the highest AUC scores obtained by both models are displayed in Figure 4.1.

	3D CNN		**3D G-CNN**	
	ReHo	VMHC	ReHo	VMHC
accuracy	0.601	0.597	0.612	0.614
sensitivity	0.498	0.470	0.619	0.516
specificity	0.696	0.701	0.580	0.721
AUC score	0.632	0.636	0.642	0.651

Table 4.3: 5-fold cross validation results of both models trained on the ReHo & VMHC summary.

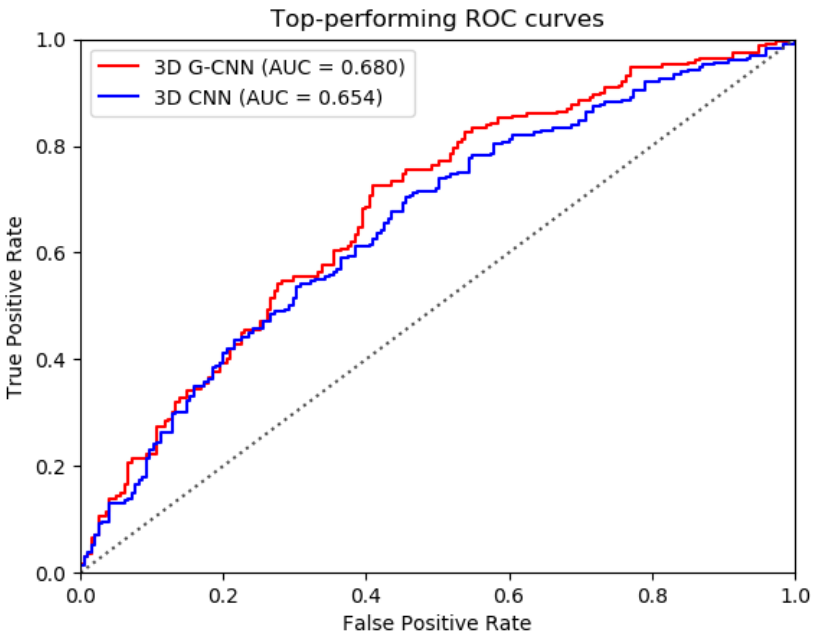


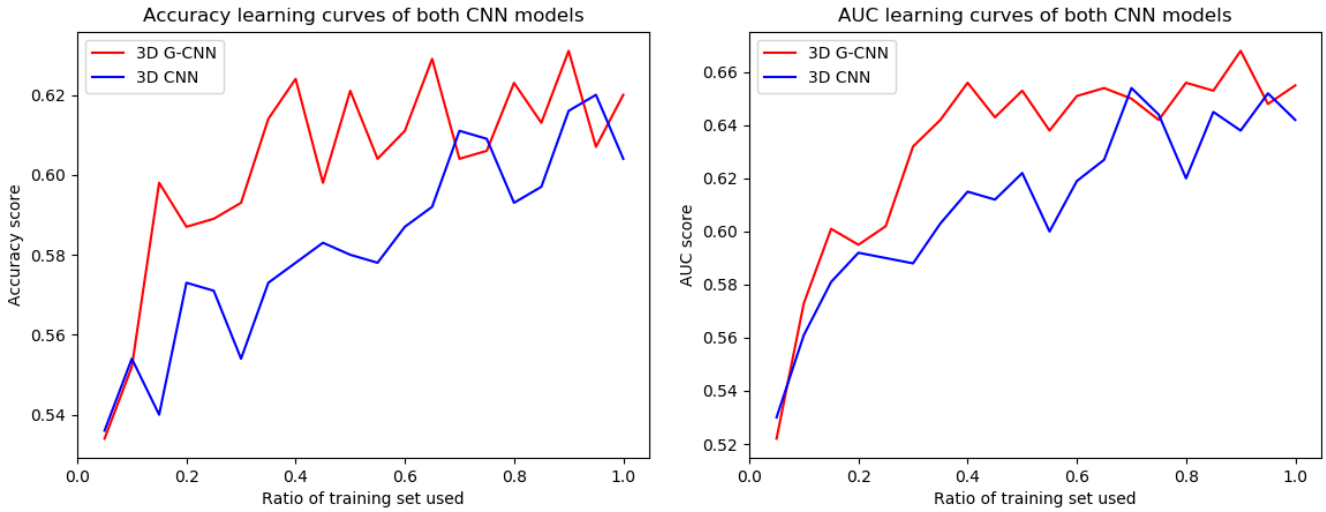
Figure 4.1: Highest AUC scores obtained by both models.

4.4 Sample complexity

The learning curves of both CNN models evaluating the effect of sample size on accuracy and AUC scores are shown in Figures 4.2a and 4.2b. In these plots, the ratio of samples used for training ranged from 5% of the ReHo training set being used (N=67) to 100% of the ReHo summary

set being used (N=1358). Moreover, both models were trained for 30 epochs on the relatively smaller training sets (ratio < 0.5) and for 50 epochs on the bigger training sets (ratio >= 0.5), because training for many epochs on smaller training sets seemed unnecessary due to earlier overfitting in smaller datasets.

From the accuracy and AUC learning curves can be concluded that for both models performance increases when the size of training set becomes larger. Notable however, is that the group-CNN seems to perform relatively better when trained on less data than the standard CNN, as the group-CNNs performance seems to be consistently good (i.e. scores that are close to the ones in Table 4.3 even when only 30-40% of the training set is used, while the standard CNN achieves good performance only after more than 60% of the training set is being used. This illustrates the capacity of the group-CNN model to perform relatively well when trained on few training samples.



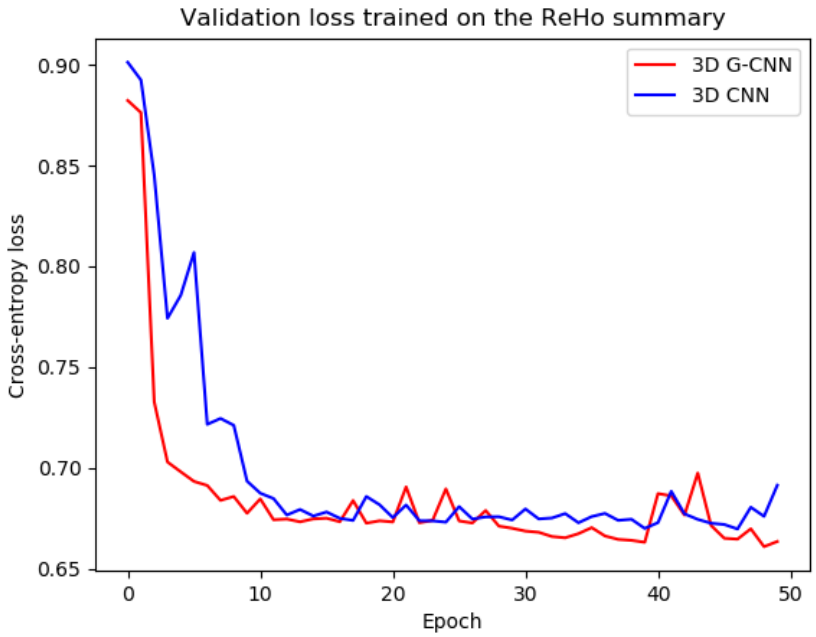
(a) Accuracy learning curves

(b) AUC learning curves

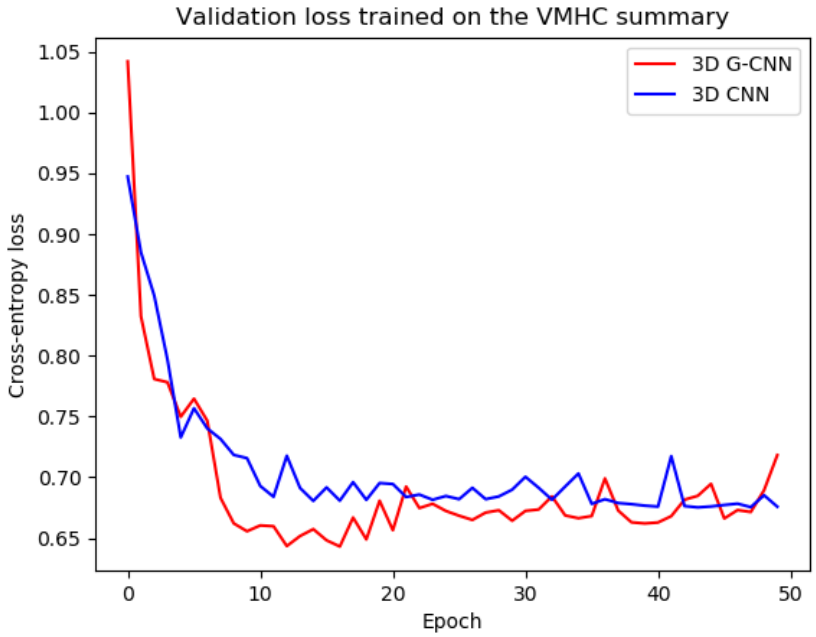
Figure 4.2: Learning curves that illustrate the impact of sample size on performance.

4.5 Loss convergence

Figure 4.3 shows the validation loss of both models plotted against the number of epochs passed in training the CNNs on the ReHo and VMHC summaries (using the whole training set). From both plots can be concluded that the group-CNN seems to converge slightly earlier than the standard CNN. More specifically, in the ReHo training process validation loss converges after approximately 10 epochs for the group-CNN, while the validation loss of the standard CNN converges after approximately 15 epochs. Regarding the VMHC summary, the group-CNN converges again after 10 epochs, whereas the standard CNN converges after approximately 20 epochs.



(a) Trained on the ReHo summary



(b) Trained on the VMHC summary

Figure 4.3: Validation loss during training plotted for both CNN models.

Chapter 5

Conclusion

The main goal of this thesis was to determine whether 3D group-convolutional neural networks perform better in classifying on ASD when trained on the ABIDE dataset than their regular 3D convolution counterparts. To test this, experiments were conducted that analyzed performance over three aspects.

Firstly, results have shown that group-CNNs achieve higher general classification performance than the standard CNNs, reporting averaged summary accuracy scores of 59.0% against 58.1%, respectively. Group-CNNs also outperformed the standard CNNs in the 5-fold cross-validation scheme when trained on the top-performing Regional Homogeneity (ReHo) and Voxel-Mirrored Homotopic Connectivity (VMHC) summaries. Regarding the ReHo summary, the group-CNNs achieved averaged accuracy/AUC scores of 61.2%/64.2 against scores of 60.1%/63.2 obtained by the standard CNN. For the VMHC summary, these results were 61.4% acc./65.1 AUC against 60.1% acc./59.7 AUC, respectively.

Secondly, results have shown the capability of group-CNNs to perform better than the standard CNNs when trained on less data. This was illustrated by the learning curves plotted for the accuracy and AUC metrics, in which the curves of the group-CNN rose to higher scores earlier than the standard CNN, with a corresponding lower ratio of used data samples. These results confirm the supposed effect of group-CNNs to reduce sample complexity, which was proposed in previous research [22, 23]. Therefore, results in this thesis emphasize once again the improvement of data efficiency that group-convolutions may have to offer, compared with regular convolutions.

Thirdly, in analyzing convergence, plotting the validation loss against the number of epochs passed in the training process revealed that the rate of convergence for the group-CNN was slightly higher than the convergence of the standard CNN. Although the differences in convergence are marginal between the two CNN models, they are meaningful, certainly because the used dataset of approximately 2000 training samples is significantly smaller than regularly used datasets in deep learning applications, meaning that the small differences in convergence here are likely to be magnified when the number of training samples increases.

Altogether, the results of the conducted experiments in this thesis have demonstrated the potential that group-CNNs may have to offer in the neuroimaging domain, by demonstrating higher overall classification performance, reduced sample complexity and higher rate of convergence than regular CNNs. Although the comparative results of the group-CNNs and the standard CNNs may not have been as powerful as the results reported in [22, 23], results of this thesis are still encouraging, and they unveil that exciting prospects may lie ahead for further implementation of group-CNNs in neuroimaging context.

Chapter 6

Discussion

As was already briefly mentioned in the conclusion, results of this thesis may not be as convincing as previous studies exploring the implementation of group-CNNs. Moreover, top performing classification accuracy results achieved in this thesis are significantly lower than cutting edge results of autism classification using the ABIDE dataset (i.e. the 73.3% acc. reported by [53]). However, the fact that the results obtained by the group-CNNs in this thesis cannot compete with the state-of-the-art results does not necessarily mean that the results of this thesis are less useful, because the main goal of this thesis was to demonstrate possible benefits of implementing group-CNNs over standard CNNs when classifying on neuroimaging data, and not to provide a classifying framework that could achieve cutting edge results in the neuroimaging domain. Nevertheless, it is useful to briefly discuss some aspects of the approach taken in this thesis that may have contributed to limitations in performance of the proposed CNN methods. Therefore, three limitations in the approach taken in this thesis will be discussed.

Firstly, the data from the ABIDE set was used without any consideration of phenotypic information of sample subjects. Moreover, classification performance might increase if subjects in the dataset are separated into subsets based on phenotypic information (e.g. age or gender) or the site location from where the MRI data was obtained, as these subdivisions of the dataset considerably reduce heterogeneity between samples. Another way to incorporate this extra dimension of information within a group-CNN framework could be the implementation a graph-based approach, in which subjects in the dataset are linked to each other based on phenotypic similarities.

Secondly, in the approach taken in this thesis both CNN models were trained and tested separately on all MRI-derived summaries, which means that not all information of the dataset was explored during each test run, but rather segments of information. Therefore, it could be that an ensemble learning strategy such as the one taken in [53] would yield better results. In such an approach, one could train a group-CNN for each summary, and subsequently use prediction scores of all summaries to generate a weighted prediction score on which can be classified. This way, information of all separate summaries is explored when classifying subjects.

Thirdly, a limitation of this thesis is the fact that only symmetry group O has been implemented and tested within the G-convolutional layers. This constraint of only exploring one symmetry group is mainly the cause of time restraints and a need for narrowing down the scope of research within this thesis. Therefore, future research could implement and test other symmetry groups too (i.e. group O_h, D_4 and D_4h), to explore and compare classification results of CNNs that are equivariant to other filter transformations, such as reflections.

As can be noted, all the limitations of this study simultaneously offer opportunities for further research. In fact, because G-convolutions can be implemented relatively easily in existing

CNN-based classification frameworks, one could essentially replace regular convolutions in top performing frameworks quite effortlessly with G-convolutions, and test if performance might increase. The main significance of this thesis is therefore to be found in the demonstration of several benefits that the use of G-convolutions can offer over regular convolutions, using only a relatively simple CNN framework. One advantage that should be highlighted in particular is the reduced sample complexity group-CNNs seem to achieve, which could be especially useful in the neuroimaging domain where data is usually scarce. To conclude, work in this thesis has only revealed the exciting potential that group-CNNs may have to offer in neuroimaging research, and it is up to future research to determine if this potential could be realized.

Bibliography

[1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, p. 436, 2015.

[2] P. Savadjiev, J. Chong, A. Dohan, M. Vakalopoulou, C. Reinhold, N. Paragios, and B. Gallix, "Demystification of ai-driven medical image interpretation: past, present and future," *European radiology*, vol. 29, no. 3, pp. 1616–1624, 2019.

[3] S. Vieira, W. H. Pinaya, and A. Mechelli, "Using deep learning to investigate the neuroimaging correlates of psychiatric and neurological disorders: Methods and applications," *Neuroscience & Biobehavioral Reviews*, vol. 74, pp. 58–75, 2017.

[4] A. S. Heinsfeld, A. R. Franco, R. C. Craddock, A. Buchweitz, and F. Meneguzzi, "Identification of autism spectrum disorder using deep learning and the abide dataset," *NeuroImage: Clinical*, vol. 17, pp. 16–23, 2018.

[5] B. Biswal, F. Zerrin Yetkin, V. M. Haughton, and J. S. Hyde, "Functional connectivity in the motor cortex of resting human brain using echo-planar mri," *Magnetic resonance in medicine*, vol. 34, no. 4, pp. 537–541, 1995.

[6] M. D. Greicius, B. Krasnow, A. L. Reiss, and V. Menon, "Functional connectivity in the resting brain: a network analysis of the default mode hypothesis," *Proceedings of the National Academy of Sciences*, vol. 100, no. 1, pp. 253–258, 2003.

[7] M. D. Fox, D. Zhang, A. Z. Snyder, and M. E. Raichle, "The global signal and observed anticorrelated resting state brain networks," *Journal of neurophysiology*, vol. 101, no. 6, pp. 3270–3283, 2009.

[8] B. B. Biswal, M. Mennes, X.-N. Zuo, S. Gohel, C. Kelly, S. M. Smith, C. F. Beckmann, J. S. Adelstein, R. L. Buckner, S. Colcombe, *et al.*, "Toward discovery science of human brain function," *Proceedings of the National Academy of Sciences*, vol. 107, no. 10, pp. 4734–4739, 2010.

[9] S. M. Smith, K. L. Miller, G. Salimi-Khorshidi, M. Webster, C. F. Beckmann, T. E. Nichols, J. D. Ramsey, and M. W. Woolrich, "Network modelling methods for fmri," *Neuroimage*, vol. 54, no. 2, pp. 875–891, 2011.

[10] R. J. Meszlényi, K. Buza, and Z. Vidnyánszky, "Resting state fmri functional connectivity-based classification using a convolutional neural network architecture," *Frontiers in neuroinformatics*, vol. 11, p. 61, 2017.

[11] P. Kassraian-Fard, C. Matthis, J. H. Balsters, M. H. Maathuis, and N. Wenderoth, "Promises, pitfalls, and basic guidelines for applying machine learning classifiers to psychiatric imaging data, with autism as an example," *Frontiers in psychiatry*, vol. 7, p. 177, 2016.

[12] D. D. Cox and R. L. Savoy, "Functional magnetic resonance imaging (fmri)"brain reading": detecting and classifying distributed patterns of fmri activity in human visual cortex," *Neuroimage*, vol. 19, no. 2, pp. 261–270, 2003.

[13] J. Mourao-Miranda, A. L. Bokde, C. Born, H. Hampel, and M. Stetter, "Classifying brain states and determining the discriminating activation patterns: support vector machine on functional mri data," *NeuroImage*, vol. 28, no. 4, pp. 980–995, 2005.

[14] Y. Fan, H. Rao, H. Hurt, J. Giannetta, M. Korczykowski, D. Shera, B. B. Avants, J. C. Gee, J. Wang, and D. Shen, "Multivariate examination of brain abnormality using both structural and functional mri," *NeuroImage*, vol. 36, no. 4, pp. 1189–1199, 2007.

[15] F. Pereira, T. Mitchell, and M. Botvinick, "Machine learning classifiers and fmri: a tutorial overview," *Neuroimage*, vol. 45, no. 1, pp. S199–S209, 2009.

[16] J. S. Anderson, J. A. Nielsen, A. L. Froehlich, M. B. DuBray, T. J. Druzgal, A. N. Cariello, J. R. Cooperrider, B. A. Zielinski, C. Ravichandran, P. T. Fletcher, *et al.*, "Functional connectivity magnetic resonance imaging classification of autism," *Brain*, vol. 134, no. 12, pp. 3742–3754, 2011.

[17] D. Zhang, D. Shen, A. D. N. Initiative, *et al.*, "Multi-modal multi-task learning for joint prediction of multiple regression and classification variables in alzheimer's disease," *NeuroImage*, vol. 59, no. 2, pp. 895–907, 2012.

[18] L. Q. Uddin, K. Supekar, C. J. Lynch, A. Khouzam, J. Phillips, C. Feinstein, S. Ryali, and V. Menon, "Salience network–based classification and prediction of symptom severity in children with autism," *JAMA psychiatry*, vol. 70, no. 8, pp. 869–879, 2013.

[19] M. Plitt, K. A. Barnes, and A. Martin, "Functional connectivity classification of autism identifies highly predictive brain features but falls short of biomarker standards," *NeuroImage: Clinical*, vol. 7, pp. 359–366, 2015.

[20] 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.

[21] A. Di Martino, D. O'connor, B. Chen, K. Alaerts, J. S. Anderson, M. Assaf, J. H. Balsters, L. Baxter, A. Beggiato, S. Bernaerts, *et al.*, "Enhancing studies of the connectome in autism using the autism brain imaging data exchange ii," *Scientific data*, vol. 4, p. 170010, 2017.

[22] T. Cohen and M. Welling, "Group equivariant convolutional networks," in *International conference on machine learning*, pp. 2990–2999, 2016.

[23] M. Winkels and T. S. Cohen, "3d g-cnns for pulmonary nodule detection," *arXiv preprint arXiv:1804.04656*, 2018.

[24] J. Kim, V. D. Calhoun, E. Shim, and J.-H. Lee, "Deep neural network with weight sparsity control and pre-training extracts hierarchical features and enhances classification performance: Evidence from whole-brain resting-state functional connectivity patterns of schizophrenia," *Neuroimage*, vol. 124, pp. 127–146, 2016.

[25] H.-I. Suk, S.-W. Lee, D. Shen, A. D. N. Initiative, *et al.*, "Deep ensemble learning of sparse regression models for brain disease diagnosis," *Medical image analysis*, vol. 37, pp. 101–113, 2017.

[26] P. Howlin, S. Goode, J. Hutton, and M. Rutter, "Adult outcome for children with autism," *Journal of child psychology and psychiatry*, vol. 45, no. 2, pp. 212–229, 2004.

[27] D. L. Christensen, K. V. N. Braun, J. Baio, D. Bilder, J. Charles, J. N. Constantino, J. Daniels, M. S. Durkin, R. T. Fitzgerald, M. Kurzius-Spencer, *et al.*, "Prevalence and characteristics of autism spectrum disorder among children aged 8 years—autism and developmental disabilities monitoring network, 11 sites, united states, 2012," *MMWR Surveillance Summaries*, vol. 65, no. 13, p. 1, 2018.

[28] A. P. Association *et al.*, *Diagnostic and statistical manual of mental disorders (DSM-5®)*. American Psychiatric Pub, 2013.

[29] R. K. Kana, L. E. Libero, and M. S. Moore, "Disrupted cortical connectivity theory as an explanatory model for autism spectrum disorders," *Physics of life reviews*, vol. 8, no. 4, pp. 410–437, 2011.

[30] N. M. Kleinhans, T. Richards, L. Sterling, K. C. Stegbauer, R. Mahurin, L. C. Johnson, J. Greenson, G. Dawson, and E. Aylward, "Abnormal functional connectivity in autism spectrum disorders during face processing," *Brain*, vol. 131, no. 4, pp. 1000–1012, 2008.

[31] C. S. Monk, S. J. Peltier, J. L. Wiggins, S.-J. Weng, M. Carrasco, S. Risi, and C. Lord, "Abnormalities of intrinsic functional connectivity in autism spectrum disorders," *Neuroimage*, vol. 47, no. 2, pp. 764–772, 2009.

[32] M. Assaf, K. Jagannathan, V. D. Calhoun, L. Miller, M. C. Stevens, R. Sahl, J. G. O'boyle, R. T. Schultz, and G. D. Pearlson, "Abnormal functional connectivity of default mode subnetworks in autism spectrum disorder patients," *Neuroimage*, vol. 53, no. 1, pp. 247–256, 2010.

[33] M. S. Cohen and S. Y. Bookheimer, "Localization of brain function using magnetic resonance imaging," *Trends in neurosciences*, vol. 17, no. 7, pp. 268–277, 1994.

[34] B. B. Biswal, "Resting state fmri: a personal history," *Neuroimage*, vol. 62, no. 2, pp. 938–944, 2012.

[35] P. C. Mulders, P. F. van Eijndhoven, A. H. Schene, C. F. Beckmann, and I. Tendolkar, "Resting-state functional connectivity in major depressive disorder: a review," *Neuroscience & Biobehavioral Reviews*, vol. 56, pp. 330–344, 2015.

[36] J. M. Sheffield and D. M. Barch, "Cognition and resting-state functional connectivity in schizophrenia," *Neuroscience & Biobehavioral Reviews*, vol. 61, pp. 108–120, 2016.

[37] M. R. Arbabshirani, K. Kiehl, G. Pearlson, and V. D. Calhoun, "Classification of schizophrenia patients based on resting-state functional network connectivity," *Frontiers in neuroscience*, vol. 7, p. 133, 2013.

[38] E. Formisano, F. De Martino, and G. Valente, "Multivariate analysis of fmri time series: classification and regression of brain responses using machine learning," *Magnetic resonance imaging*, vol. 26, no. 7, pp. 921–934, 2008.

[39] A. Abraham, M. P. Milham, A. Di Martino, R. C. Craddock, D. Samaras, B. Thirion, and G. Varoquaux, "Deriving reproducible biomarkers from multi-site resting-state data: an autism-based example," *NeuroImage*, vol. 147, pp. 736–745, 2017.

[40] M. J. Rosa, L. Portugal, T. Hahn, A. J. Fallgatter, M. I. Garrido, J. Shawe-Taylor, and J. Mourao-Miranda, "Sparse network-based models for patient classification using fmri," *Neuroimage*, vol. 105, pp. 493–506, 2015.

[41] F. Liem, G. Varoquaux, J. Kynast, F. Beyer, S. K. Masouleh, J. M. Huntenburg, L. Lampe, M. Rahim, A. Abraham, R. C. Craddock, *et al.*, "Predicting brain-age from multimodal imaging data captures cognitive impairment," *Neuroimage*, vol. 148, pp. 179–188, 2017.

[42] J. Bernal, K. Kushibar, D. S. Asfaw, S. Valverde, A. Oliver, R. Martí, and X. Lladó, "Deep convolutional neural networks for brain image analysis on magnetic resonance imaging: a review," *Artificial intelligence in medicine*, 2018.

[43] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255, Ieee, 2009.

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

[45] M. Nieuwenhuis, N. E. van Haren, H. E. H. Pol, W. Cahn, R. S. Kahn, and H. G. Schnack, "Classification of schizophrenia patients and healthy controls from structural mri scans in two large independent samples," *Neuroimage*, vol. 61, no. 3, pp. 606–612, 2012.

[46] C. J. Brown, J. Kawahara, and G. Hamarneh, "Connectome priors in deep neural networks to predict autism," in *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)*, pp. 110–113, IEEE, 2018.

[47] A. Di Martino, C.-G. Yan, Q. Li, E. Denio, F. X. Castellanos, K. Alaerts, J. S. Anderson, M. Assaf, S. Y. Bookheimer, M. Dapretto, *et al.*, "The autism brain imaging data exchange: towards a large-scale evaluation of the intrinsic brain architecture in autism," *Molecular psychiatry*, vol. 19, no. 6, p. 659, 2014.

[48] S. G. Mueller, M. W. Weiner, L. J. Thal, R. C. Petersen, C. R. Jack, W. Jagust, J. Q. Trojanowski, A. W. Toga, and L. Beckett, "Ways toward an early diagnosis in alzheimer's disease: the alzheimer's disease neuroimaging initiative (adni)," *Alzheimer's & Dementia*, vol. 1, no. 1, pp. 55–66, 2005.

[49] M. P. Milham, D. Fair, M. Mennes, S. H. Mostofsky, *et al.*, "The adhd-200 consortium: a model to advance the translational potential of neuroimaging in clinical neuroscience," *Frontiers in systems neuroscience*, vol. 6, p. 62, 2012.

[50] S. Liu, S. Liu, W. Cai, S. Pujol, R. Kikinis, and D. Feng, "Early diagnosis of alzheimer's disease with deep learning," in *2014 IEEE 11th international symposium on biomedical imaging (ISBI)*, pp. 1015–1018, IEEE, 2014.

[51] G. Lee, K. Nho, B. Kang, K.-A. Sohn, and D. Kim, "Predicting alzheimer's disease progression using multi-modal deep learning approach," *Scientific reports*, vol. 9, no. 1, p. 1952, 2019.

[52] A. Ortiz, J. Munilla, J. M. Gorriz, and J. Ramirez, "Ensembles of deep learning architectures for the early diagnosis of the alzheimer's disease," *International journal of neural systems*, vol. 26, no. 07, p. 1650025, 2016.

[53] M. Khosla, K. Jamison, A. Kuceyeski, and M. R. Sabuncu, "3d convolutional neural networks for classification of functional connectomes," in *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*, pp. 137–145, Springer, 2018.

[54] L. Zou, J. Zheng, C. Miao, M. J. Mckeown, and Z. J. Wang, "3d cnn based automatic diagnosis of attention deficit hyperactivity disorder using functional and structural mri," *IEEE Access*, vol. 5, pp. 23626–23636, 2017.

[55] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, *et al.*, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[56] S. B. Kotsiantis, I. Zaharakis, and P. Pintelas, "Supervised machine learning: A review of classification techniques," *Emerging artificial intelligence applications in computer engineering*, vol. 160, pp. 3–24, 2007.

[57] Y. LeCun, Y. Bengio, *et al.*, "Convolutional networks for images, speech, and time series," *The handbook of brain theory and neural networks*, vol. 3361, no. 10, p. 1995, 1995.

[58] Stanford University, "Cs231n: Convolutional neural networks for visual recognition." `http://cs231n.stanford.edu/`. Accessed on: 14-05-2019.

[59] "Convolutional neural networks (cnn): step 1 - convolution operation." `https://www.superdatascience.com/`, 2018. Accessed on: 16-06-2019.

[60] D. Scherer, A. Müller, and S. Behnke, "Evaluation of pooling operations in convolutional architectures for object recognition," in *International conference on artificial neural networks*, pp. 92–101, Springer, 2010.

[61] B. Boehmke and B. Greenwell, "Hands-on machine learning with r." `https://bradleyboehmke.github.io/HOML/index.html`, 2019. Accessed on: 24-06-2019.

[62] 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.

[63] M. R. Arbabshirani, S. Plis, J. Sui, and V. D. Calhoun, "Single subject prediction of brain disorders in neuroimaging: promises and pitfalls," *Neuroimage*, vol. 145, pp. 137–165, 2017.

[64] J. M. Mateos-Pérez, M. Dadar, M. Lacalle-Aurioles, Y. Iturria-Medina, Y. Zeighami, and A. C. Evans, "Structural neuroimaging as clinical predictor: A review of machine learning applications," *NeuroImage: Clinical*, 2018.

[65] S. Wager, S. Wang, and P. S. Liang, "Dropout training as adaptive regularization," in *Advances in neural information processing systems*, pp. 351–359, 2013.

[66] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[67] P. Luo, X. Wang, W. Shao, and Z. Peng, "Towards understanding regularization in batch normalization," *International Conference on Learning Representations (ICLR)*, 2018.

[68] S. Ji, W. Xu, M. Yang, and K. Yu, "3d convolutional neural networks for human action recognition," in *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pp. 495–502, Citeseer, 2010.

[69] "Invariant principle." `https://brilliant.org/wiki/invariant-principle-definition/`. Accessed on: 28-05-2019.

[70] R. Kondor and S. Trivedi, "On the generalization of equivariance and convolution in neural networks to the action of compact groups," *arXiv preprint arXiv:1802.03690*, 2018.

[71] G. Cheng, P. Zhou, and J. Han, "Rifd-cnn: Rotation-invariant and fisher discriminative convolutional neural networks for object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2884–2893, 2016.

[72] K. Lenc and A. Vedaldi, "Understanding image representations by measuring their equivariance and equivalence," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 991–999, 2015.

[73] P. Y. Simard, D. Steinkraus, J. C. Platt, *et al.*, "Best practices for convolutional neural networks applied to visual document analysis.," in *Icdar*, vol. 3, 2003.

[74] S. Dieleman, J. De Fauw, and K. Kavukcuoglu, "Exploiting cyclic symmetry in convolutional neural networks," *arXiv preprint arXiv:1602.02660*, 2016.

[75] Y. Zhang, M. Brady, and S. Smith, "Segmentation of brain mr images through a hidden markov random field model and the expectation-maximization algorithm," *IEEE transactions on medical imaging*, vol. 20, no. 1, pp. 45–57, 2001.

[76] S. M. Smith, "Fast robust automated brain extraction," *Human brain mapping*, vol. 17, no. 3, pp. 143–155, 2002.

[77] C. Craddock, S. Sikka, B. Cheung, R. Khanuja, S. S. Ghosh, C. Yan, Q. Li, D. Lurie, J. Vogelstein, R. Burns, *et al.*, "Towards automated analysis of connectomes: The configurable pipeline for the analysis of connectomes (c-pac)," *Front Neuroinform*, vol. 42, 2013.

[78] Z. Yu-Feng, H. Yong, Z. Chao-Zhe, C. Qing-Jiu, S. Man-Qiu, L. Meng, T. Li-Xia, J. Tian-Zi, and W. Yu-Feng, "Altered baseline brain activity in children with adhd revealed by resting-state functional mri," *Brain and Development*, vol. 29, no. 2, pp. 83–91, 2007.

[79] Q.-H. Zou, C.-Z. Zhu, Y. Yang, X.-N. Zuo, X.-Y. Long, Q.-J. Cao, Y.-F. Wang, and Y.-F. Zang, "An improved approach to detection of amplitude of low-frequency fluctuation (alff) for resting-state fmri: fractional alff," *Journal of neuroscience methods*, vol. 172, no. 1, pp. 137–141, 2008.

[80] M. R. Arbabshirani, E. Damaraju, R. Phlypo, S. Plis, E. Allen, S. Ma, D. Mathalon, A. Preda, J. G. Vaidya, T. Adali, *et al.*, "Impact of autocorrelation on functional connectivity," *Neuroimage*, vol. 102, pp. 294–308, 2014.

[81] X.-N. Zuo, R. Ehmke, M. Mennes, D. Imperati, F. X. Castellanos, O. Sporns, and M. P. Milham, "Network centrality in the human functional connectome," *Cerebral cortex*, vol. 22, no. 8, pp. 1862–1875, 2011.

[82] "c-pac 1.4.3 beta documentation." `https://fcp-indi.github.io/docs/user/index.html`. Accessed on: 15-06-2019.

[83] D. Tomasi and N. D. Volkow, "Functional connectivity density mapping," *Proceedings of the National Academy of Sciences*, vol. 107, no. 21, pp. 9885–9890, 2010.

[84] M. O. Sokunbi, R. T. Staff, G. D. Waiter, T. S. Ahearn, H. C. Fox, I. J. Deary, J. M. Starr, L. J. Whalley, and A. D. Murray, "Inter-individual differences in fmri entropy measurements in old age," *IEEE Transactions on Biomedical Engineering*, vol. 58, no. 11, pp. 3206–3214, 2011.

[85] D. J. Wang, K. Jann, C. Fan, Y. Qiao, Y.-F. Zang, H. Lu, and Y. Yang, "Neurophysiological basis of multi-scale entropy of brain complexity and its relationship with functional connectivity," *Frontiers in neuroscience*, vol. 12, p. 352, 2018.

[86] Y. Zang, T. Jiang, Y. Lu, Y. He, and L. Tian, "Regional homogeneity approach to fmri data analysis," *Neuroimage*, vol. 22, no. 1, pp. 394–400, 2004.

[87] M. J. Hoptman, X.-N. Zuo, D. D'Angelo, C. J. Mauro, P. D. Butler, M. P. Milham, and D. C. Javitt, "Decreased interhemispheric coordination in schizophrenia: a resting state fmri study," *Schizophrenia research*, vol. 141, no. 1, pp. 1–7, 2012.

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

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

[90] P. Jain, P. Kar, *et al.*, "Non-convex optimization for machine learning," *Foundations and Trends in Machine Learning*, vol. 10, no. 3-4, pp. 142–336, 2017.

[91] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv preprint arXiv:1609.04747*, 2016.

[92] R. Parikh, A. Mathai, S. Parikh, G. C. Sekhar, and R. Thomas, "Understanding and using sensitivity, specificity and predictive values," *Indian journal of ophthalmology*, vol. 56, no. 1, p. 45, 2008.

[93] E. J. Boyko, "Ruling out or ruling in disease with the most sensitive or specific diagnostic test: Short cut or wrong turn?," *Medical Decision Making*, vol. 14, no. 2, pp. 175–179, 1994.

[94] J. Huang and C. X. Ling, "Using auc and accuracy in evaluating learning algorithms," *IEEE Transactions on knowledge and Data Engineering*, vol. 17, no. 3, pp. 299–310, 2005.

[95] T. Fawcett, "An introduction to roc analysis," *Pattern recognition letters*, vol. 27, no. 8, pp. 861–874, 2006.

Appendix A

Setup

The available hardware for training both CNNs concerns a CPU with Intel E-5-2640 v4 (running on 10 cores/20 threads), and two GPUs for calculation: (1) NVidia GeForce GTX 1080 Ti (12 GB memory) and NVidia Titan XP (12GB memory). Using these GPUs, training for 100 epochs typically lasted only 10 minutes for the standard CNN, and roughly 30 minutes for the group-CNN. Moreover, data was formatted in hdf5 format and the CNNs were implemented using the tensorflow deep learning library. An overview of all the used libraries and corresponding versions can be found in Table A.1.

Library	**Version**
cudatoolkit	9.0
cudnn	7.3.1
h5py	2.9.0
hdf5	1.10.4
matplotlib	1.5.3
numpy	1.16.3
python	3.6.8
tensorflow-gpu	1.13.1

Table A.1: Overview of used libraries in technical implementation.