Machine Learning Approaches in Imaging Genetics

Karianne Tesaker

A thesis presented for the degree of Master in Informatics

Department of Informatics
University of Bergen

Supervisor: Tom Michoel

Norway

June 1, 2021

Acknowledgements

I would like to acknowledge and thank the following individuals and groups for their contributions and support.

My supervisor Prof. Tom Michoel for all his guidance, contributions, ideas, read-throughs and answering of all my questions.

PhD-candidate Muhammad Ammar Malik, for his contributions, ideas and answering of questions.

Assoc. Prof. Alexander Selvikvåg Lundervold for his contributions and for providing pre-processed MRI brain scans used in preparation for this thesis.

The Alzheimer's Disease Neuroimaging Initiative (ADNI) (National Institutes of Health Grant U01 AG024904) and DOD ADNI (Department of Defense award number W81XWH-12-2-0012) for the funding of data collection and sharing for this project. ADNI is funded by the National Institute on Aging, the National Institute of Biomedical Imaging and Bioengineering, and through generous contributions from the following: AbbVie, Alzheimer's Association; Alzheimer's Drug Discovery Foundation; Araclon Biotech; BioClinica, Inc.; Biogen; Bristol-Myers Squibb Company; CereSpir, Inc.; Cogstate; Eisai Inc.; Elan Pharmaceuticals, Inc.; Eli Lilly and Company; EuroImmun; F. Hoffmann-La Roche Ltd and its affiliated company Genentech, Inc.; Fujirebio; GE Healthcare; IXICO Ltd.; Janssen Alzheimer Immunotherapy Research & Development, LLC.; Johnson & Johnson Pharmaceutical Research & Development LLC.; Lumosity; Lundbeck; Merck & Co., Inc.; Meso Scale Diagnostics, LLC.; NeuroRx Research; Neurotrack Technologies; Novartis Pharmaceuticals Corporation; Pfizer Inc.; Piramal Imaging; Servier; Takeda Pharmaceutical Company; and Transition Therapeutics. The Canadian Institutes of Health Research is providing funds to support ADNI clinical sites in Canada. Private sector contributions are facilitated by the Foundation for the National Institutes of Health (www.fnih.org). The grantee organization is the Northern California Institute for Research and Education, and the study is coordinated by the Alzheimer's Therapeutic Research Institute at the University of Southern California. ADNI data are disseminated by the Laboratory for Neuro Imaging at the University of Southern California.

My fellow students through my years at the University of Bergen and University of Stavanger, for their support and company, especially during long days at the libraries and study halls in times of especially heavy work load. Especially thanks to Tuva Torblå Augedal, Anita Lien, Natalie P. Wannaphong and Anastasia Mathisen for their encour-

agement and company this last year.

My friends and family, above all my parents, Anne Vik Såghus and Ken Henry Tesaker, for their support, encouragement and understanding.

Finally, I want to express my gratitude to Odd Inge Nerland, for his love and support.

Thank you,

Karianne Tesaker
Bergen, 29.05.2021

Abstract

Established approaches in imaging genetics and genome wide association studies (GWAS) such as univariate, multivariate and voxel-wise approaches, are prone to certain disadvantages such as being computationally expensive, selection of imaging phenotypes (IPs) requiring knowledge of which features are relevant for the task, and/or that relationships between different IPs are lost. In this thesis, uses of Random Forest Classification (RFC) and Convolutional Neural Networks (CNNs) within imaging genetic studies of magnetic resonance imaging (MRI) and genetic data from the Alzheimer's Disease Neuroimaging Initiative (ADNI) database, have been investigated, with the hope of addressing the issues of the established approaches. CNNs were found to be a possible powerful tool in assessing which brain areas are affected by specific single nucleotide polymorphisms (SNPs).

Abbreviations

2D	Two dimensional
3D	Three dimensional
Aβ	Amyloid beta peptides
ABCA7	Adenosine triphosphate–binding cassette subfamily A member 7
AD	Alzheimer's disease
ADNI	Alzheimer's Disease Neuroimaging Initiative
AI	Artificial intelligence
ANN	Artificial neural network
ANT	Advanced Normalization Tools
APOE	Apolipoprotein E
APP	Amyloid precursor protein
BIN1	Bridging integrator 1
BN	Batch Normalization
CD2AP	CD2-associated protein
CF	Confounder Filtering
CLU	Clusterin
CN	Normal control subjects
CNN	Convolutional neural network
CR1	Complement receptor 1
DL	Deep learning
ECE-1	Endothelin-converting enzyme-1
EMCI	Early mild cognitive impairment
FVGWAS	Fast Voxelwise Genome Wide Association analysiS

GD	Gradient descent
GWAS	Genome-wide association study
ICA	Independent component analysis
IP	Image phenotype
IVA	Independent vector analysis
LMCI	Late mild cognitive impairment
MCI	Mild cognitive impairment
MDR	Multifactor dimensionality reduction
ML	Machine learning
MRI	Magnetic resonance imaging
NMR	Nuclear magnetic resonance
NN	Neural network
NU	Non-Uniform intensity normalization
PCA	Principal component analysis
PET	Positron emission tomography
PICALM	Phosphatidylinositol binding clathrin assembly protein
RF	Random forest
RFC	Random forest classifier
RFE	Recursive feature elimination
RFECV	Recursive feature elimination and cross-validated selection
ReLU	Rectified Linear Unit function
ROI	Region of interest
SGD	Stochastic gradient descent
SMC	Significant memory concerns
tanh	Hyperbolic tangent function
VGWAS	Voxel-wise genome-wide association study

Contents

Acknowledgements i

Abstract iii

Abbreviations iv

Contents vi

List of Figures x

List of Tables xiii

1 Introduction 1
1.1 Aims . . . 3

2 Background 4
2.1 An Introduction to Machine Learning . . . 4
2.1.1 Training ML Models . . . 4
2.1.1.1 Overfitting and Underfitting . . . 4
2.1.1.2 Bias and Variance . . . 5
2.1.1.3 Model Selection . . . 5
2.1.2 Evaluating Performance of Classification Models . . . 6
2.2 Random Forest Classification . . . 7
2.2.1 Ensemble Methods . . . 7
2.2.1.1 Bagging . . . 8
2.2.2 Decision Trees . . . 8
2.2.2.1 Regularization for Decision Trees . . . 10
2.2.3 Random Forest . . . 10
2.3 The Problem of High Dimensional Data . . . 10
2.3.1 Feature Selection . . . 10
2.3.1.1 Random Forest Feature Importance . . . 11
2.3.1.2 Permutation Feature Importance . . . 11
2.3.1.3 Recursive Feature Elimination . . . 11
2.3.2 Feature Extraction . . . 11
2.4 Deep Learning . . . 12
2.4.1 Activation Functions . . . 13
2.4.2 Gradient Descent . . . 14
2.4.3 Back-Propagation . . . 14

2.4.4 Cost Functions . . . 15
2.4.5 The Adam Optimizer . . . 15
2.4.6 Regularization for Deep Learning . . . 15
2.4.7 Batch Normalization . . . 16
2.4.8 Transfer Learning . . . 16
2.5 Convolutional Neural Networks . . . 17
2.5.1 Convolutions . . . 17
2.5.2 Convolutional Layers . . . 18
2.5.3 Pooling Layers . . . 20
2.5.4 DenseNet . . . 20
2.5.5 Occlusion Sensitivity . . . 22
2.6 Alzheimer's Disease and Magnetic Resonance Imaging . . . 22
2.6.1 The Human Brain . . . 22
2.6.1.1 Cerebrum . . . 22
2.6.1.2 Cerebellum . . . 24
2.6.1.3 Brain Stem . . . 24
2.6.2 Magnetic Resonance Imaging . . . 25
2.6.3 Alzheimer's Disease . . . 25
2.6.3.1 Visibility of AD on MRI . . . 26
2.7 Genetics . . . 26
2.7.1 Genome-Wide Association Studies and Imaging Genetics . . . 26
2.7.2 Genetic Risk Factors of AD . . . 27
2.7.3 Machine Learning in Imaging Genetics . . . 27
2.8 Statistics . . . 28
2.8.1 *P*-value Hypothesis Testing . . . 28
2.8.2 Permutation Tests . . . 29
2.8.2.1 Permutation Tests as a Way of Estimating Classifier Performance . . . 30
2.8.3 Multiple Hypothesis Testing . . . 30
2.8.3.1 False Discovery Rate . . . 30

3 Methods **31**
3.1 Materials . . . 31
3.1.1 Study Population . . . 31
3.1.2 MRI Data Preprocessing for Phase 1: the RFC Tasks . . . 32
3.1.3 MRI Data Preprocessing for Phase 2: the DL Tasks . . . 34
3.1.4 Genetic Data . . . 35
3.2 Python Modules . . . 35
3.2.1 Scikit-learn . . . 35
3.2.2 ELI5 . . . 35
3.2.3 Statsmodels . . . 35
3.2.4 PyTorch . . . 36
3.2.5 MONAI . . . 36
3.3 Phase 1: Random Forest Classifier in Imaging Genetics . . . 36
3.3.1 Feature Vectors . . . 36
3.3.2 Getting Started: Classification of Number of APOE ε4 Alleles . . . 37
3.3.3 Feature Selection Using RF Feature Importance . . . 38

3.3.4 Using Selected Feature Dataset for RF Classification of APOE ε4 Alleles . . . 39
3.3.5 GWAS 1: *P*-Value Estimation of 305,479 SNPs . . . 39
3.3.5.1 Criteria for Data Filtering . . . 39
3.3.5.2 *P*-Value Estimation 1: Ranking by Accuracy Test Score 39
3.3.5.3 *P*-Value Estimation 2: Ranking by Ratio between Accuracy Test Score and Majority Class Representation . . . 40
3.3.5.4 *P*-Value Estimation 3: Permutation Test Scores . . . 40
3.3.6 GWAS 2: *P*-value estimation of SNPs Located at Genes Connected to AD . . . 40
3.3.6.1 First Permutation Feature Importance Feature Selection . . . 41
3.3.6.2 Second Permutation Feature Importance Feature Selection . . . 41
3.3.7 APOE ε4 Carrier Status Classification . . . 42
3.4 Phase 2: Deep Learning in Imaging Genetics . . . 42
3.4.1 APOE ε4 Carrier Status Classification with DenseNet . . . 42
3.4.2 3D DenseNet Classification of rs11193198 and rs2243454 . . . 43
3.4.3 Occlusion Sensitivity . . . 44

4 Results and Analysis **46**
4.1 Phase 1: Random Forest Classifier in Imaging Genetics . . . 46
4.1.1 Getting Started: Classification of Number of APOE ε4 Alleles . 46
4.1.2 Feature Selection Using RF Feature Importance . . . 48
4.1.3 Using Selected Feature Dataset for RF Classification of APOE ε4 Alleles . . . 51
4.1.4 GWAS 1: *P*-Value Estimation of 305,479 SNPs . . . 52
4.1.4.1 *P*-Value Estimation 1 and 2: Ranking by Accuracy Test Score and by Ratio between Accuracy Test Score and Majority Class Representation . . . 52
4.1.4.2 *P*-Value Estimation 3: Permutation Test Scores . . . 54
4.1.5 GWAS 2: *P*-value estimation of SNPs Located at Genes Connected to AD . . . 56
4.1.5.1 The Feature Selections . . . 56
4.1.5.2 The Permutation Tests . . . 58
4.1.6 APOE ε4 Carrier Status Classification . . . 61
4.2 Phase 2: Deep Learning in Imaging Genetics . . . 63
4.2.1 APOE ε4 Carrier Status Classification with DenseNet . . . 63
4.2.2 3D DenseNet Classification of rs11193198 and rs2243454 . . . 66
4.2.3 Occlusion Sensitivity . . . 67

5 Discussion and Concluding Remarks **71**
5.1 Phase 1: Random Forest Classifier in Imaging Genetics . . . 71
5.2 Phase 2: Deep Learning in Imaging Genetics . . . 73
5.3 Conclusion . . . 77

Bibliography **78**

Appendix 1 - Tables for Methods **89**

Appendix 2 - Occlusion Sensitivity Results **98**

List of Figures

2.1 Illustration of an underfitted (left), a well-fitted (middle) and an overfitted (right) model to the data [84]. 5
2.2 Example of a confusion matrix with three class labels, C1, C2 and C3 . 6
2.3 Example of a dataset and the corresponding decision tree [7]. 9
2.4 Illustration of simple feedforward neural network with two hidden layers [85]. 12
2.5 Simple illustration of a single artificial neuron [85]. 13
2.6 Three common activation functions for artificial neural networks [37]. . 13
2.7 Example of a CNN, classifying images into four categories: dog, cat, boat and bird [85]. 17
2.8 Example of a 2D convolution with output restricted to positions where the kernel lies entirely within the image. Input of dimensions 4×3 with kernel of dimensions 2×2 yield output of dimensions 3×2 [45]. 18
2.9 Simple illustration of convolutional layer of six $5 \times 5 \times 3$ filters [85]. . . 19
2.10 Example of max pooling with 2×2 filters. **Stride** refer to how many locations (along the x or y axis) to shift before computing new output [85]. 20
2.11 Occlusion maps with different mask sizes for CNN predicting "tiger cat" category. 22
2.12 Regions of the human brain and location of some brain functions [114]. 23
2.13 The human brain anatomy [94]. 23
2.14 The limbic lobe and surrounding structures [19]. 24
2.15 MR images of healthy brain (left) and AD brain (right) [10]. 26

3.1 Distributions of gender and APOE4 over the diagnostic groups for the two datasets used in phase 1 and phase 2, respectively 32
3.2 Distribution APOE $\varepsilon 4$ class labels for the dataset 38
3.3 Distribution of rs11193198 and rs2243454 class labels per subject. . . . 43

4.1 Confusion matrices for the dummy classifier and each RFC model trained with different sets of features from feature vector 3, classification of APOE $\varepsilon 4$. 47
4.2 Top and bottom left: Permutation test scores for each RFC model trained with different sets of features from feature vector 3, in classification of APOE $\varepsilon 4$. Bottom right: RF feature importances for the RFC model with the full dataset. 47
4.3 Confusion matrices for each RFC model, classification of APOE $\varepsilon 4$. . . 48
4.4 RFECV results for RFCs trained on the given datasets. 49

4.5 Random Forest Feature Importance for the selected features of feature vector 8 and 10. 50
4.6 Results from hyper-parameter tuning of RFC model for binary classification of APOE ε4. 51
4.7 Random Forest Feature Importance for Classification of APOE ε4 . . . 52
4.8 Accuracy test score distributions and Manhattan plots for binary and 3-class SNPs, *p*-values estimated using rank based on accuracy. 53
4.9 Permutation test score *p*-values for binary SNPs, plotted against their respective accuracy test score or ratio ranking. 53
4.10 Ratio distributions and Manhattan plots for binary and 3-class SNPs, *p*-values estimated using rank based on ratio between accuracy test scores and majority class representation. 54
4.11 The distribution and Manhattan plot of permutation test *p*-values for all 305,479 SNPs. 55
4.12 The distribution of permutation test *p*-values for SNP Dataset 2 and the distribution per chromosome of all SNPs that achieved the best possible *p*-value score of 0.009901. 55
4.13 The distribution of permutation test *p*-values for SNP Dataset 3 and the distribution per chromosome of all SNPs that achieved the best possible *p*-value score of 0.009901. 56
4.14 The results from the RFECV of the first feature selection (making Dataset X), and the permutation feature importances for the 15 RFE selected features from the second feature selection (making Dataset Y). 56
4.15 Pairwise Pearson correlation coefficients for all features of Dataset X and Dataset Y. The labels on the x-axis are the same as on the y-axis. . . 57
4.16 P-Value distributions from the permutation tests of Dataset X and Dataset Y. 58
4.17 Heatmaps of permutation feature importances for top scoring SNPs of their respective distributions. 60
4.18 Heatmaps of permutation feature importances of common features in Dataset X and Dataset Y for the top scoring SNPs of Dataset X. 61
4.19 Heatmaps of permutation feature importances of common features in Dataset X and Dataset Y for the top scoring SNPs of Dataset Y. 61
4.20 Confusion matrix for RF classification of APOE ε4 carrier status using feature Dataset Y. 62
4.21 Both RF and permutation feature importances for RF classification of APOE ε4 carrier status using feature Dataset Y. 62
4.22 Train loss and validation accuracy per epoch for all models of the APOE ε4 pretraining pipelines. (Read from left) first row: Gender model, AD-1 model; second row: AD-2 model, AD-3 model; third row: APOE-1 model, APOE-2 model; fourth row: APOE-3, APOE-4. The left axes of the graphs give loss, while the right axes give accuracy. 64
4.23 Confusion matrices for the DenseNet-121 models from the four pretraining pipelines. 65
4.24 Train loss and validation accuracy per epoch for the two SNP model pipelines. From left: rs11193198 model, rs2243454 model. 66

4.25 Confusion matrices for the two SNP model pipelines. From left: rs11193198 model, rs2243454 model. 67
4.26 The effect of mask size and stride size on occlusion sensitivity. To the left is the brain image slice of the occlusion maps. The mask size and stride size coupling will be given as *a*/*b*, where *a* gives the $a \times a \times a$ mask size, and *b* gives the $b \times b \times b$ stride size. From the left: 8/8, 16/4, 16/8, 16/16, 24/4, 24/16, 32/32. 67
4.27 Occlusion sensitivity maps for AD-1 DenseNet model (5-way classification of AD, no pretraining), for prediction of label "EMCI". To the left is the image slice, to the right is the occlusion map of this image slice. Slices have been made along the following axis (from top): x=60, y=60, y=50 and y=40. For the definition of the axes, see figure A.2.1 of Appendix 2. 68

List of Tables

2.1 DenseNet architectures. The growth rate for all the networks is k=32. Each "conv" layer in the table corresponds to the BN-ReLU-Conv sequence described [53]. 21

3.1 Distribution of the study population into the diagnostic study groups for the data used for the RF and DL classification tasks. 31
3.2 Description of feature vectors 37

4.1 Accuracy test scores for the RFC models. Feature vector number gives which dataset was used for training the model (see table 3.2). 46
4.2 Accuracy test, cross validation and p-value scores for each RFC model trained with different sets of features from feature vector 3, in addition to the accuracy test score of the DummyClassifier in classification of APOE ε4. 46
4.3 The number of features selected from RFECV and accuracy test scores for RF classification of AD health status, of models trained with the selected features and all features for the given feature vector numbers (see table 3.2). 49
4.4 Data on all SNPs achieving p-value less than 0.05 in permutation tests of Dataset X, ordered from best to worst. Columns 1. p-value and 1. cross-val score indicate the p-value and cross validation accuracy score from this permutation tests of Dataset X, respectively. Columns 2. p-value and 2. cross-val score indicate the p-value and cross validation accuracy score from the latter permutation tests of Dataset Y, for comparison. The differences in p-value and cross validation scores are given as scores from Dataset Y subtracted from scores from Dataset X. . 58
4.5 Data on all SNPs achieving p-value less than 0.05 in permutation tests of Dataset Y, ordered from best to worst. Columns 2. p-value and 2. cross-val score indicate the p-value and cross validation accuracy score from this permutation tests of Dataset Y, respectively. Columns 1. p-value and 1. cross-val score indicate the p-value and cross validation accuracy score from the former permutation tests of Dataset X, for comparison. The differences in p-value and cross validation scores are given as scores from Dataset Y subtracted from scores from Dataset X. . 59
4.6 DenseNet-121 model descriptions. 63
4.7 Test scores for the DenseNet121 classification models. For model description, see table 4.6. For binary classification, precision, recall and F1 scores are listed in addition to accuracy. 66

5.1 *P*-value results from stage 1 and 2 of the third *p*-value estimation through permutation testing from GWAS1 for SNPs associated with hippocampal, amygdala and/or cortical volume [12; 54]. 71
5.2 *P*-value results from GWAS1 and GWAS2 of SNPs listed as the polymorphism of the AlzGene Database top 42 genetic risk factors, that exist in the datasets. GWAS1-3.1 and GWAS1-3.2 refer to stage 1 and 2 of the third *p*-value estimation through permutation testing from GWAS1. GWAS2-X and GWAS2-Y refer to the GWAS2 *p*-value distribution X and Y, respectively. 72
5.3 Test performance of the three individual models of the 3D DenseNet ensemble for 4-way classification of AD diagnosis from study performed by Ruiz et al. in 2020 [98]. 74

1 Introduction

Imaging genetics is an emerging interdisciplinary research field, aiming to assess, discover and evaluate the associations between genetic variations and imaging measures [119]. In other words, imaging genetic studies provides the opportunity to further our understanding of the impact of genetic variation on individual differences in behaviour, personality, disease development and body function. A common kind of genetic variation are single nucleotide polymorphisms (SNPs). A SNP is a variation in nucleotide base at a single position in a DNA sequence that are quite common between individuals of the same species [78]. *Genome-wide association studies* (GWAS) investigates associations between numerous SNPs and phenotypes [118].

The first imaging genetic studies used linear regression in univariate analysis of association between a selected few candidate regions of interest (ROI) in the brain and genetic markers [80]. By performing a great number of pair-wise univariate analyses, this method can be used for brain-wide genome-wide studies. Voxel-wise approaches fits separate models to each image voxel, often for multiple genetic markers at the time, also analysing the relationship between genetic markers. However, such multiple univariate analysis are highly computationally expensive, and faces several challenges with regard to finite CPU speed, and limited CPU nodes and computer memory [54]. For n genetic variants and m brain locations, $n \cdot m$ analyses are performed. With all known genetic variants known today, and all possible brain image locations, $n \cdot m$ can easily be a total of $\sim 10^{12}$.

To address the computational disadvantages of voxel-wise GWAS (VGWAS), Huang et al. presented the Fast Voxelwise Genome Wide Association AnalysiS (FVGWAS) framework in 2015 [54]. With this framework they were able to assess 501,584 SNPs with 193,275 voxels in a little over two days on a single CPU [54].

Commonly, several areas of the brain are involved in carrying out brain functions. Thus, another disadvantage of massive univariate and voxel-wise approaches is that this relationship between the brain areas, or more generally, the *image phenotypes* (IPs), are lost [80]. Multivariate high-dimensional regression to entire datasets have been employed to tackle this issue. At the same time, such multivariate analysis is also highly computationally expensive with high-dimensional data. Thus, brain image information is commonly summarized using a relatively small number of brain summary measures across som key ROIs [80]. Selecting such regions of interest requires previous knowledge of which are wise to choose for the task at hand. Consequently, information from unknown related regions is lost. Another approach is feature extraction methods , such as principal component analysis (PCA). In addition to PCA, common tools used in

multivariate analysis, both for assessing multiple image features and multiple genetic markers, include multifactor dimensionality reduction (MDR), independent component analysis(ICA), and clustering [72].

A problem with the feature extraction dimensionality reduction approaches, is that model interpretability, i.e. finding which IPs specific genetic markers are associated with, is lost. Random Forest (RF) is a machine learning (ML) algorithm that can process high-dimensional data fast, performing feature selection implicitly [76]. Such speedy algorithms could be answer to many of the challenges of the traditional imaging genetic approaches.

As magnetic resonance imaging (MRI) can perform measures of brain anatomy and function, the symptoms of neurological disorders such as Alzheimer's disease (AD) and Schizophrenia are visible in MRI data [27]. Signatures showing AD in MRI data include decrement of brain tissue volumes, firstly in the medial temporal lobe, and abnormalities in neuronal activity while performing cognitive tasks [57]. Understanding and discovering genetic risk factors of such diseases is an important step for development of practices in prevention, diagnosis and treatment of them [54]. Alzheimer's Disease Neuroimaging Initiative (ADNI), a study aimed to develop biomarkers that helps with early detection and tracking of AD [8], and other big studies and research projects collecting medical image and genetic data from large cohorts, have become a great resource for imaging genetic studies aimed to assess and discover genetic variants associated with the diseases.

For traditional ML classification and regression, brain features are typically engineered from MRI scans by segmenting the brain structure into anatomical regions or tissue types, and then calculating different characteristics such as region volumes and tissue thickness [90]. Deep learning (DL) algorithms such as convolutinal neural networks (CNNs, however, can work with raw image data directly. Due to their great ability at learning useful representations of images, CNNs are of great interest in medical imaging. These image features automatically learned by a CNN during training, have long surpassed the features engineered by hand [73]. DL is therefore becoming a methodology of choice for analyzing medical images, and has seen applications within image classification, disease diagnosis, object detection, image segmentation, image restoration, image reconstruction and other tasks [73; 105].

The ADNI has collected thousands of brain scans, genetic profiles, and blood and cerebrospinal fluid biomarkers from almost two thousand elderly individuals that either are control subjects or have mild cognitive impairment (MCI) or AD. The MRI datasets from ADNI have been used to train and validate CNNs before. One such model was designed to predict MCI subjects that will convert to AD [71]. The model was boosted with assisted structural brain image features and achieved an accuracy of 79.9%. Another CNN, also designed to predict MCI conversion to AD, as well as predicting the individual diagnosis of AD, achieved an accuracy of 99% in predicting AD, and 75% in discriminating between MCI which will convert to AD and stable MCI [14].

1.1 Aims

The aim of this study was to research the uses of Random Forest Classification (RFC) and CNNs in imaging genetics, hopefully addressing some of the disadvantages of established approaches. Firstly, by using classification of AD diagnosis for feature selection of hand engineered features from MRI data. Secondly, by assessing genetic variation associated with AD, in performing brain-wide genome-wide studies using RFC with the selected features. Lastly, by using the power of CNNs to find important data structures, to assess which brain areas are affected by a selected set of genetic variants, directly on raw MR image data. The MRI, diagnostic and genetic data from the ADNI database have been used for these studies.

2 Background

2.1 An Introduction to Machine Learning

Machine learning (ML) is a subfield of *artificial intelligence* (AI), whose basic principle is automatic modeling of underlying processes that have generated the collected data [64]. A machine learning algorithm produces a model based on the data. Mitchell [77] defined *learning* as:

> *"A computer program is said to **learn** from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.*

Machine learning is split into three types: *supervised, unsupervised* and *reinforcement learning*. In supervised learning, the data is labeled, meaning the algorithm is fed the "answer" to the objects during training. Supervised learning is used for classification and regression tasks. In classification, labels are categorical. An example of a classification task is to classify animals from images. In regression, labels are continuous. An example of a regression task is to predict age of individuals from images. Performance measure of supervised learning is typically prediction accuracy or error. Unsupervised learning uses no labels in it's learning process. Typical unsupervised tasks are clustering and dimensionality reduction. In reinforcement learning there are no object/label pairs, only feedback from the actions of an agent in an environment. The agent tries to maximize a reward by its actions. The result of a single move is not important, but the result of a sequence of moves [2]. The agent learns from previous actions sequences. A typical example is game playing, like a chess AI.

This thesis will focus on supervised learning and classification problems.

2.1.1 Training ML Models

2.1.1.1 Overfitting and Underfitting

The ultimate goal of machine learning is *generalization*, making (good) predictions about unseen data. A complicated model will usually fit the training data better than a simple model, however, these models don't necessarily generalize well. A complicated model is often *overfitted* to the training, while a simple model is often *underfitted*. Overfitting is when the model learns noise as well as the underlying function of the data [4]. Underfitting is when the learned function is less complicated than the underlying function. In simple terms, an overfitted model performs well on training data but gen-

eralizes poorly, an underfitted model both perform badly on training data as well as on unseen data. Overfitting, underfitting and a good estimation of the underlying function is illustrated in figure 2.1. Summing this up, simple, but not too simple, models generalizes better than complex models, which corresponds to the principle *Occam's razor*, that states *"(...) simpler explanations are more plausible and any unnecessary complexity should be shaved off."* [4]

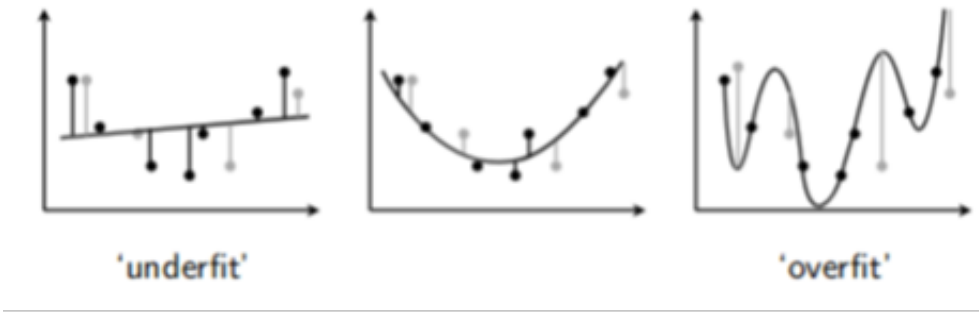


Figure 2.1: Illustration of an underfitted (left), a well-fitted (middle) and an overfitted (right) model to the data [84].

2.1.1.2 Bias and Variance

Bias and variance are two kinds of prediction errors in machine learning. Bias is error due to modelling assumptions, variance is error due to variations in the training set. Total prediction error is composed of bias and variance [65]. A big goal for good ML modelling is to minimize both bias and variance, however, lowering one usually increases the other. This is known as the bias/variance dilemma. A complex model will generally fit better to the underlying function, thus bias decreases. However, small fluctuations in the dataset will then have a greater affect on the fitting of the model, thus variance increases. To sum up, underfitting models have bias, and overfitting models have variance [5]. To some extent, overfitting and underfitting can be reduced by having more data, as it gives the ML algorithm more cases to learn the underlying function. In machine learning, one can never have too much data. Another kind of prediction error is called irreducible error, which is due to the noise in the data. Irreducible error can't be reduced except by data cleaning. Using the animal classification of images example from section 2.1, a way of data cleaning would be cropping out the animals from the background.

2.1.1.3 Model Selection

To get an unbiased estimation of a model's generalization performance, data must be split into separate test and train sets before fitting. Furthermore, to select the best model for the task, i.e. selecting the model with the best balance between bias and variance, the data can be split three ways: into train, validation and test sets. Validation data is unseen data used to select a model, while test data is unseen data used to evaluate the performance of the selected model. In model selection, both different ML algorithms and different hyper-parameters of these respective algorithms, can be tested by training each model instance, then measuring their performance on the validation data. The

model with the best performance on the validation data, i.e. the model that generalized best, is selected.

Another method for model selection is cross validation. The principle of cross validation is estimating the quality of the final model (theory) as an average of different models (theories) [65]. In *leave-one-out cross validation*, each object from the train set is left out, and the remaining objects are used for learning. Performance on the left out object (validation object) is measured, and this step is repeated for all objects of the train set. The validation performance is then the average of all the individual performances. However, this process can be very time consuming. In *k-fold cross validation*, the train set is split into k approximately equally sized subsets [65]. Performance is then estimated for each of these subsets by using the remaining subsets as learning sets, just like for the leave-one-out method. *Stratified k-fold cross validation* is similar, only the subsets are build so as to preserve the original class distribution of the full set [65]. Cross validation can also be used as a final performance estimation of a model, which is especially useful if the total number of objects in the dataset is so small that leaving out a test sets hinders learning.

2.1.2 Evaluating Performance of Classification Models

Classification accuracy and confusion matrix are common methods to evaluate performance of classifiers [65]. Classification accuracy measures the success of a classifier by estimating the relative frequency of correct classifications

$$accuracy = \frac{n_{corr}}{n} \tag{2.1}$$

where n_{corr} equals the number of correct predictions and n is the total number of predictions. Typically, classification accuracy is given in percent ($accuracy \cdot 100\%$). A confusion matrix displays the frequencies of class predictions verses the correct class label. The diagonal of the matrix gives n_{corr} per class. Figure 2.2 shows an example of a confusion matrix. This example confusion matrix shows that the model has some problems with separating classes C1 and C2, but clearly sees the distinction between them and class C3.

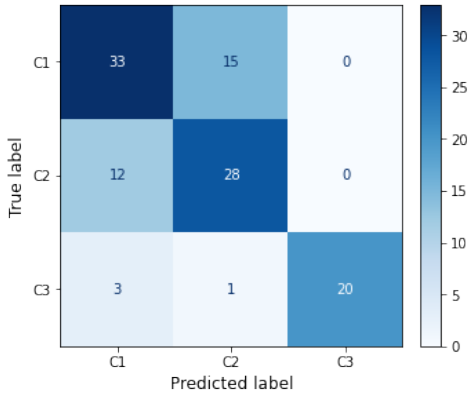


Figure 2.2: Example of a confusion matrix with three class labels, C1, C2 and C3

When the class label distribution for the dataset is very unbalanced, other performance measures than accuracy may be more informative. *Precision, recall* and *F1 score* are examples of such performance measures. These measures in first instance only works with binary (two-class) problems, though they can be tweaked to work with multi-class problems. Let us first define one class as *true*, this will be the minority class, and the other as *false*, the majority class, for the binary classification problem. Precision and recall use the four elements of the confusion matrix of binary classifications: *true positives* (TP), *false positives* (FP), *true negatives* (TN) and *false negatives* (FN). Precision estimates the portion of true positives amongst all positives, mathematically

$$precision = \frac{TP}{TP+FP}. \tag{2.2}$$

Recall, also called sensitivity, is the relative frequency of correctly classified positives [65], mathematically

$$recall = \frac{TP}{TP+FN}. \tag{2.3}$$

Simply put, precision yields information of the validity of the results, while recall informs how complete they are. If one does not want to miss a single possible positive instance, recall is a good performance measure to use. An example could be finding people at risk of an illness; the consequences would be possibly greater of wrongly putting someone out of the risk zone, than wrongly putting someone into the risk zone. If, on the other hand, wrongly classifying positive instances is a greater problem, precision is a good performance measure. An example of this could be classifying spam email; classifying an important email as spam could potentially lead to the email account user never reading it. The F1 score is an in-between of precision and recall, if both are important for the classification task, this is a good measure to use. F1 score, defined as

$$F1 = \frac{2 \cdot precision \cdot recall}{precision + recall}\ , \tag{2.4}$$

is the harmonic mean of precision and recall [65]. Computing precision and/or recall for each class, then compute the mean, can be one way to use these measures for multi-class problems. However, much of the original advantages of these methods will then be lost.

2.2 Random Forest Classification

2.2.1 Ensemble Methods

In machine learning, ensemble methods are collections of classifiers/regressors where the final prediction of the ensemble is either made by voting or a mean probability vector. The individual models of an ensemble are called *base-learners* [3]. Each base-learner can be weak and simple with large error, the only criteria being that they are

better than random. A great property of the ensemble method, is that as long as the base-learners are independent, it reduces variance without affecting bias. Thus, the performance of the ensemble can be very good, even though each individual model is weak, as long as the base-learners have low bias. There are many ways to make the base-learners independent, the simplest being that each are trained with different data. Other ways are using different learning algorithms, using different hyper-parameters if it's the same learning algorithm, or using different seeds for randomized algorithms.

2.2.1.1 Bagging

Bagging is the act of randomly drawing training subsets from the full training set, to train each base-learner of the ensemble, then take the average over the base-learners to make predictions [3]. For an ensemble of L base-learners, the L train subsets needed are generated by a method called *bootstrapping*. The L train subsets, x_i for $i \in [1, L]$, of size n, are drawn from the full train set X of size N with *replacement*. A weakness of this method is that some samples of X may be drawn many times, while some will not be drawn at all [3]. An *unstable algorithm* is a learning algorithm highly affected by small changes in the train set, i.e. with high variance. Bagging works best with unstable algorithms for base-learners.

2.2.2 Decision Trees

A *decision tree* is an unstable supervised learning algorithm [3], simply put, it is a set of decision rules. It consists of *a root, internal nodes, edges* and *leaves*. The root is the topmost node, the nodes correspond to attributes, the edges to subsets of attribute values, while leaves are terminal nodes corresponding to class labels [66]. Each path from root to leaf is a decision rule. The tree is called a regression tree if the leaf labels are continuous. At each node, a test function with discrete outcomes is applied to the input, and one of the branches are taken depending on the outcome [7]. The process starts at the root node, and is recursively repeated until a leaf node, which yields the output. The optimal tree is the smallest possible tree.

In a *univariate tree*, the test function at each node uses only one of the input dimensions (i.e. features) [7]. If the feature has n discrete values (e.g. colour, gender, etc.), then it will generate a n-way split, each branch corresponding to each discrete value. If the feature has continuous numerical values, the test function f_m will be a comparison. For input x_j

$$f_m(\mathbf{x}) : x_j > w_{m0} \tag{2.5}$$

where w_{m0} is a suitably chosen threshold value [7].

Tree learning algorithms are greedy, looking for the best splits from the root to the leaves. For classification trees, splits are decided using *impurity measure*. A pure split is when all inputs choosing the same branch belong to the same class, for all branches of the split [7]. Two common impurity measures are *entropy* and *gini index*. Let the probability of of class C_i be given by

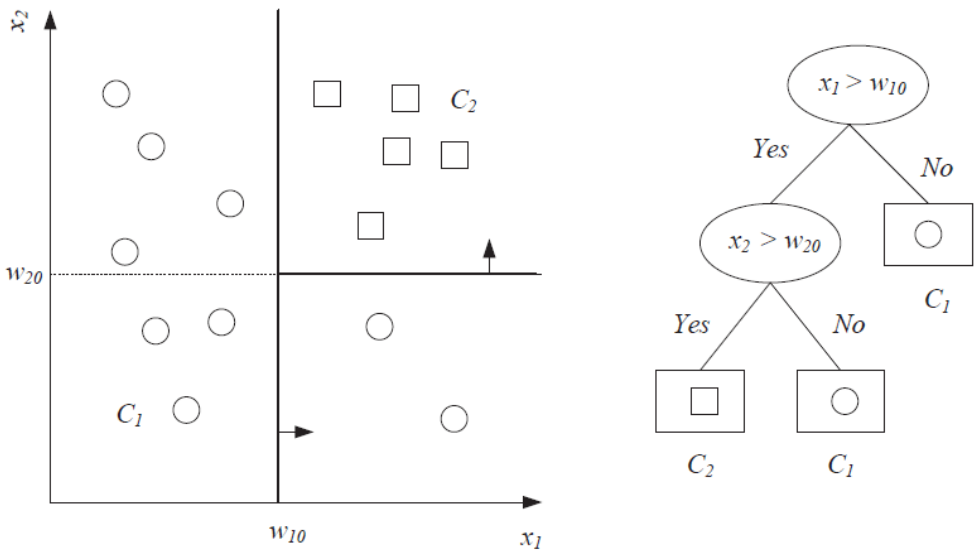


Figure 2.3: Example of a dataset and the corresponding decision tree [7].

$$p_m^i = \frac{N_m^i}{N_m} \tag{2.6}$$

where N_m^i is the number of training instances that reaches node m and N_m^i is the number of these instances that belong to class C_i. Entropy is then defined as

$$I_m = -\sum_{i=1}^{K} p_m^i log_2 p_m^i \tag{2.7}$$

where K is the total number of classes and $0log0 = 0$, while the gini index is defined as

$$I_m = \sum_{i=1}^{K} p_m^i (1 - p_m^i). \tag{2.8}$$

Node m is pure if p_m^i is either 0 or 1 for all i. If node m is not pure, the instances should be split such that impurity after the split is minimized [7]. Though this is locally optimal in order to ensure a small tree, it gives no guarantee of making the smallest possible tree. Total impurity after a split is given by

$$I'_m = \sum_{j=1}^{n} \frac{N_{mj}}{N_m} \cdot I_{mj} \tag{2.9}$$

where N_{mj} is the number of instances of N_m that selects branch j, n is the number of possible branches, and I_{mj} is the impurity measure (gini index or entropy) given by equation 2.8 or 2.9 where p_m^i is replaced by the probability of class C_i given outcome j:

$$p_{mj}^i = \frac{N_{mj}^i}{N_{mj}}. \tag{2.10}$$

2.2.2.1 Regularization for Decision Trees

Regularization techniques are strategies to decrease generalization error of a model, often at the expense of training error [43]. In other words, techniques to avoid overfitting. One such strategy to limit variance in decision trees is *prepruning*. Simply put, this means that when the number of training instances reaching a node is smaller than a set threshold, the node is not split further and thus becomes a leaf, setting the highest represented class of the instances as output. Another method is *postpruning*. The postpruning process is to backtrack the tree to try to find and prune unnecessary subtrees [7]. From the initial train set, a *pruning set* is set aside. The tree is then learned until all leaves are pure on the remaining set. Then, each subtree is recursively replaced by a leaf labeled with the majority class of the training instances, and performance on the pruning set is measured. If a new leaf node does not decrease performance, the subtree is pruned and the leaf node is kept. Other methods to limit variance include setting a maximum depth or number of leaves to the tree. Setting maximum depth means to set limit to number of nodes from root to leaf. For both maximum depth and maximum number of leaves, the tree is generally grown in a best-first fashion.

2.2.3 Random Forest

A random forest (RF) is an ensemble of decision trees. The RF utilizes the bagging method with a twist: in addition to the regular bootstrap sampling, a subset of the features of the train set is also randomly selected (with repetition), meaning that the subsets have lower dimensionality than the full set. Dropping bootstrapping, letting each tree use the full train set, though only a limited set of features from the train set, is also possible [65].

2.3 The Problem of High Dimensional Data

As stated before, noise is responsible for the irreducible error of an ML model. Generally, high dimensional data contain much noise, as many features may be completely irrelevant for the task. High dimensional data also make learning computationally expensive. This section will discuss many methods of cleaning high dimensional input data, divided into two categories, *feature selection* and *feature extraction*.

2.3.1 Feature Selection

Feature selection methods involve selecting the most useful features in the dataset for the given task, and ignoring the rest. In *subset selection*, the goal is to find the best subset of the set of features, meaning the one that most contributes to accuracy [6]. There are 2^d possible subsets of d features, so unless d is small, it is not possible to test for all. Therefore, to perform feature selection, a *measure of the importance* of each feature with regard to the task at hand, is very useful. Though, there exists algorithms for finding the best selection of features without it. *RF feature importance* and *permutation feature importance* are examples of such measures.

2.3.1.1 Random Forest Feature Importance

The RFC performs feature selection *implicitly*, by using only a small subset of features for the classification. The feature importance scoring is a by-product of the training of the RFC and provides a relative ranking of the features [76]. Also known as the Gini importance, RF feature importance are *impurity-based*. They are each computed as the normalized total reduction of the impurity measure criterion brought by that feature. More explicitly, the decrease in impurity for the optimal splits at each node in each tree in the RF, is recorded and accumulated individually for each feature [76]. This quantity indicates how often a particular feature was selected for a split, and how large its overall impurity reduction was. However, RF feature importances have been found to not be reliable for all datasets. This is due to the RF feature importance computation being biased, favoring features with great variations in their scale of measurement or number of categories [83].

2.3.1.2 Permutation Feature Importance

Permutation feature importance, also known as *mean score decrease*, computes importance as a measure of decrease in model performance when the feature is permuted (i.e. becomes noise) [83]. Firstly, model performance with original dataset is measured as baseline comparison. This is either measured with a train/validation split, or with cross validation. Next, each feature is permuted n times with decrease in performance measured for each permutation, and the mean for each feature is computed and stored. Thus, positive importance indicates that the performance decreased when the feature became noise, meaning the feature was important for the predictions, while negative importance means the performance increased, thus the feature was not important. Permutation importance is more reliable than RF importance [83]. However, this algorithm also have its weakness: correlated features will yield smaller decreases in performance than they necessarily should, due to the learner having access to them in their correlated partners.

2.3.1.3 Recursive Feature Elimination

Recursive feature elimination (RFE) is an instance of *backward feature selection* [46]. In backward feature selection, one start with the full dataset, then eliminate one (or more) features at the time [6]. In RFE, the given learner is first trained on the full datset, secondly, the ranking of all features based on some importance measure is computed, finally, the feature (or features) with of the smallest ranking is removed. This procedure is then repeated for the new, smaller dimensional dataset, either until the desired dimension is reached, or until there is only one feature left, yielding a feature ranking for the dataset.

2.3.2 Feature Extraction

In feature extraction, the original features are used to create new features. The goal is to represent the important information of the data with few features. Feature extraction methods can be both supervised and unsupervised. Examples of unsupervised

feature extraction methods include *principal component analysis* (PCA), *factor analysis* and *mutlidimensional scaling*. The supervised method *linear discriminant analysis* is among the most known and widely used feature extraction methods, along with PCA [6].

2.4 Deep Learning

Deep learning (DL), or *artificial neural networks* (ANNs), are learning algorithms within machine learning that imitates the learning process of the brain. ANNs *"mimic biological neural networks by abstracting neural functions to a simple element (neuron), only able to summarize its input and normalize its output"* [64]. DL algorithms have grown massively in popularity the last decades, and are the current approach to many problems within computer vision, language modeling and robotics [73].

The principle of DL is simple; instead of manually engineering features, the algorithms learn complex features in the data themselves by passing the input through layers of non-linear transformations, the learned features are then finally passed trough a (simple) classifier or regression model. The first layer of an ANN is called the *input layer*, while the final layer of classification or regression is the *output layer*. The layers in between, the layers of non-linear transformations, or neurons, are called *hidden layers*. In *feedforward* DL models, the output of the former layer becomes the input of the next [42]. A simple model is shown in figure 2.4. At each neuron, each input is multiplied by a corresponding *weight*, the sum of all products are then fed through an *activation function* (the non-linear transformation), and the output is passed on to the next layer [85]. This neuron structure is illustrated in figure 2.5. To understand how DL algorithms work, explanations of some basic concepts are necessary.

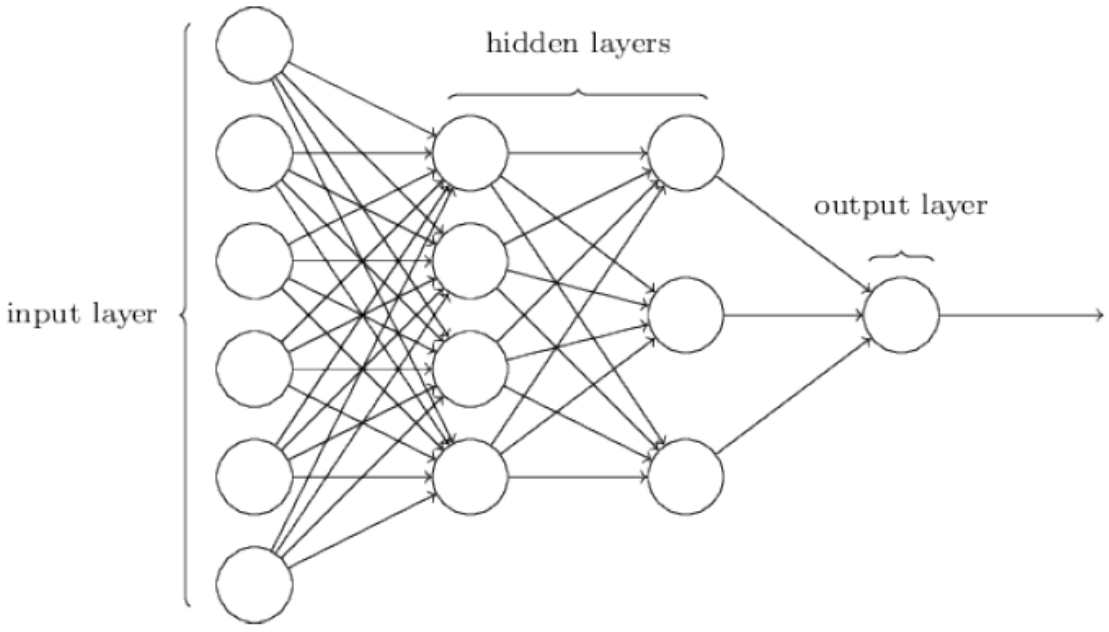


Figure 2.4: Illustration of simple feedforward neural network with two hidden layers [85].

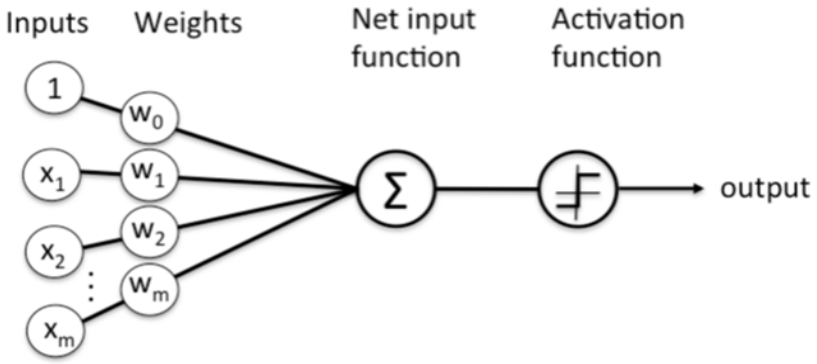


Figure 2.5: Simple illustration of a single artificial neuron [85].

2.4.1 Activation Functions

As stated earlier, activation functions yield the non-linear transformations to the input of the neurons of ANNs. Three common activation functions are *Sigmoid, Rectified Linear Unit* (ReLU) and *hyperbolic tangent function* (tanh) [37; 73], illustrated in figure 2.6. The current default recommendation is to use ReLU [42]. Mathematically, Sigmoid is defined as

$$\sigma(x) = \frac{1}{1+e^{-x}}, \tag{2.11}$$

tanh as

$$tanh(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}, \tag{2.12}$$

and ReLU as

$$f(x) = \max(0,x). \tag{2.13}$$

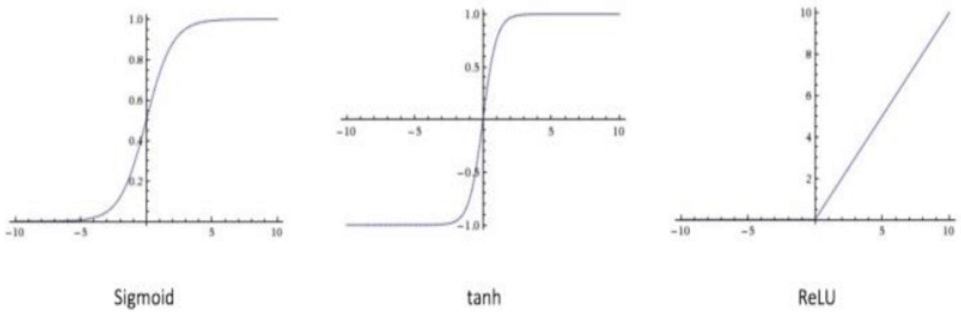


Figure 2.6: Three common activation functions for artificial neural networks [37].

The *SoftMax* activation function is often used in the output layer of neural networks (NNs). For classification, the output layer will have as many nodes as there are classes, and the SoftMax activation function yields the probability of each class. The Sigmoid function can be viewed as as the SoftMax function for two classes. SoftMax is defined as

$$\text{softmax}(x_i) = \frac{e^{x_i}}{\sum_{j=0}^{n} e^{x_j}}. \tag{2.14}$$

for $i \in [0, n)$ [40]. In the case of the output layer, n will be the number of classes, and $\text{softmax}(x_i)$, the output for node x_i, yields the probability of class C_i. SoftMax is prone to underflow or overflow with too small or too large values of x_i. Therefore $\text{softmax}(\mathbf{z})$ is often evaluated instead, where $\mathbf{z} = \mathbf{x} - \max_i x_i$ [40].

2.4.2 Gradient Descent

Most DL algorithms are powered by *gradient descent* (GD) [41]. GD is an optimization algorithm used to find a minimum value of a function. The function we want to minimize is called the *objective function*, *criterion*, *cost function*, *loss function* or *error function* [40]. The algorithm operates by taking iterative steps in the direction of steepest descent, as defined by the negative of the gradient, until a local or global minimum is reached. The cost function of DL algorithms can be multidimensional and have many local minima that are not optimal, which makes optimization difficult [40]. Therefore, the algorithm settles for finding a very low value, that is not necessarily the formally minimal value. The learning rate is a positive scalar that determines the step size [40]. Smaller learning rates are computationally expensive, leading to slower convergence, while larger learning rates can lead to too large steps, making the algorithm miss the minimum. Simply put, the gradient tells where, and the learning rate how far, to go at each iteration.

Using the whole dataset to compute the gradient at each step is computationally expensive and therefore slow [41]. A variant of the algorithm computing approximations of the gradient instead, called *stochastic gradient descent* (SGD) is therefore preferred. SGD uses randomly selected subsets, so-called *mini-batches*, of a given *batch size*, to estimate the gradient at each iteration. SGD is able to find a very low value of the cost function quickly enough to be useful [41]. For good optimization it is important that the mini-batches are selected randomly during training [44].

2.4.3 Back-Propagation

Training of DL models is an iterative process, consisting of *forward* and *backward passes* through the layers of the network. The goal of training is to optimize the network's weights, or *parameters*, to minimize a given cost function. During the forward pass, training input, typically a small batch of training points, is passed through the layers, recording the output and local derivatives at each neuron, until it reaches the output layer, and makes its prediction. The cost function is then used to measure the difference between the true outputs and the predicted outputs [73]. The backward pass, or the *back-propagation* algorithm, propagates the information of the cost backwards through the network, in order to compute the gradient [42]. In this process, the gradient of the cost function with respect to the weights in each neuron is calculated using the chain rule and dynamic programming, and the weights are updated using gradient descent [73]. An *epoch* is finished when all batches of training data have been back propagated, and all weights have been updated accordingly. Typically, the process is repeated more than once during training, i.e. for a given number of epochs.

2.4.4 Cost Functions

Typical cost functions for DL networks include *mean absolute error*, *mean squared error*, *cross-entropy loss* and *Dice loss* [73]. This section will provide a description of cross-entropy loss, which is more popular than mean absolute error and mean squared error due to them often yielding poor results when used with gradient-based optimization [42]. Cross-entropy loss is very similar to the entropy impurity measure for decision trees, discussed in section 2.2.2. The cross-entropy between the training data and the model distribution is equivalent to the negative log-likelihood. Mathematically,

$$J(\theta) = -E_{\mathbf{x},\mathbf{y}\ P_{data}} \log p_{model}(\mathbf{y}|\mathbf{x},\theta) \tag{2.15}$$

where θ is the parameter (weight) vector, $E_{\mathbf{x},\mathbf{y}\ P_{data}}$ is the expected values with respect to the data distribution (true values), and $p_{model}(\mathbf{y}|\mathbf{x},\theta)$ is the model distribution (model predictions) [42].

2.4.5 The Adam Optimizer

There have been created many variations to stochastic optimization [26; 62; 89; 116], one such variation is the Adam optimization algorithm. The name is derived from *adaptive moment estimation*, and is developed to combine the advantages of optimization algorithms *AdaGrad* and *RMSProp* [62]. Adam makes use of an exponentially decaying average of past gradients (m_t), making the algorithm converge faster, and an exponentially decaying average of past squared gradients (v_t). The hyper-parameters $\beta_1, \beta_2 \in [0,1)$ control the exponential decay rates of these two moving averages. These moving averages are initialized as zeros, making the the moment estimates biased to zero, so bias-corrected estimates, $\hat{v}_t$ and $\hat{m}_t$, are made [62]. The parameters, θ_t (at time t) is updated by the following rule:

$$\theta_t \leftarrow \theta_{t-1} - \frac{\alpha \cdot \hat{m}_t}{\sqrt{\hat{v}_t} + \varepsilon}, \tag{2.16}$$

with set hyper-parameters ε, β_1, β_2 and the learning rate α [62]. Typically, $\varepsilon = 10^{-8}$, $\beta_1 = 0.9$, $\beta_2 = 0.99$ and $\alpha = 0.001$.

2.4.6 Regularization for Deep Learning

Weight decay, *dropout*, and *early stopping* are common regularization techniques for DL algorithms. Weight decay penalizes complexity by adding a penalty to the weights of the objective function [43]. The most common forms of weight decay are *L1* and *L2 regularization*. Both methods scale the penalty by a positive hyperparameter α [43]. While L1 regularization uses the absolute sum of the parameters as penalty, L2 regularization uses the square, thereby penalizing larger weights more.

Early stopping is a simple technique that prohibits overfitting by monitoring train and validation error while training. Every time the validation error is improved, the parameters at the time is stored. If validation error has not improved for some time, the learning process is terminated, and the model is returned to the parameters at the best

validation error [43].

The strategy of simply ignoring neurons at a given probability, is known as dropout. It is computationally inexpensive, yet a powerful method of regularization, especially for ensembles of DL models [43]. In most DL algorithms, this is simply done by multiplying the output at a neuron by zero at the given dropout probability. This random removal of neurons during training results in weights of the trained network having been tuned based on optimization of multiple variations of the network [73].

2.4.7 Batch Normalization

Batch normalization (BN) is an algorithm used in optimizing DL networks, though it is not an optimization algorithm [44]. It is a method of adaptive reparametrization, helping with lessening the difficulty of training very deep models.

In practice, when training DL models, all layers are updated simultaneously, but the gradient update of each parameter is under the assumption that the other functions (of other layers) remain constant. The reparametrization of BN significantly reduces the problem of coordinating updates across many layers [44]. BN can be applied to any input or layer in a network. The normalization is defined in the following way:

$$\mathbf{H}' = \frac{\mathbf{H} - \mu}{\sigma} \tag{2.17}$$

where $\mathbf{H}$ is a minibatch of activations of the layer, arranged as a design matrix, μ is a vector containing the mean of each unit and σ is a vector containing the standard deviation at each unit. The rest of the network operate on $\mathbf{H}'$ the exact same way the original network would have operated on $\mathbf{H}$. At training time,

$$\mu = \frac{1}{m}\sum_i \mathbf{H}_{i,:} \tag{2.18}$$

and

$$\sigma = \sqrt{\delta + \frac{1}{m}\sum_i (\mathbf{H} - \mu)_i^2}, \tag{2.19}$$

where δ is a small positive value to avoid undefined gradients [44]. Through backpropagation the mean and standard deviation are computed and applied to normalize $\mathbf{H}$. BN makes it so that the gradient will never propose an operation that acts simply to increase the standard deviation or mean of h_i.

2.4.8 Transfer Learning

Strategies involving training models on other tasks before training the desired model to perform the desired task are collectively known as *pretraining* [44]. One such approach is known as *transfer learning*. In transfer learning, the learned parameters of one model, e.g. an ANN classifying vehicles in images, are transferred to a new model, e.g. an ANN classifying animals in images, with the goal of improving generalization for the new model [39]. Simply put, the idea is that it is better to start with something similar,

than starting from scratch. It could be that the input for the model is similar (or the same), or that the outputs are similar. The ANN examples of image classification tasks are examples of the first case; for these tasks, one could re-use the exact same model, only changing the output layer. In the latter case, the last layers of the networks are usually kept; i.e. in speech recognition systems, the earlier layers near the input need to learn user-specific pronunciations of the same words, while the outputs are the same for all [39].

2.5 Convolutional Neural Networks

Convolutional neural networks (CNNs), are deep learning networks that are powerful tools for learning useful representations of images and other structured data [73]. The name stems from the networks employing a specialized kind of linear operation called *convolution*. "Convolutional networks are simply neural networks that use convolution in place of general matrix multiplication in at least one of their layers" [45]. Almost all CNNs also contain *pooling* layers.

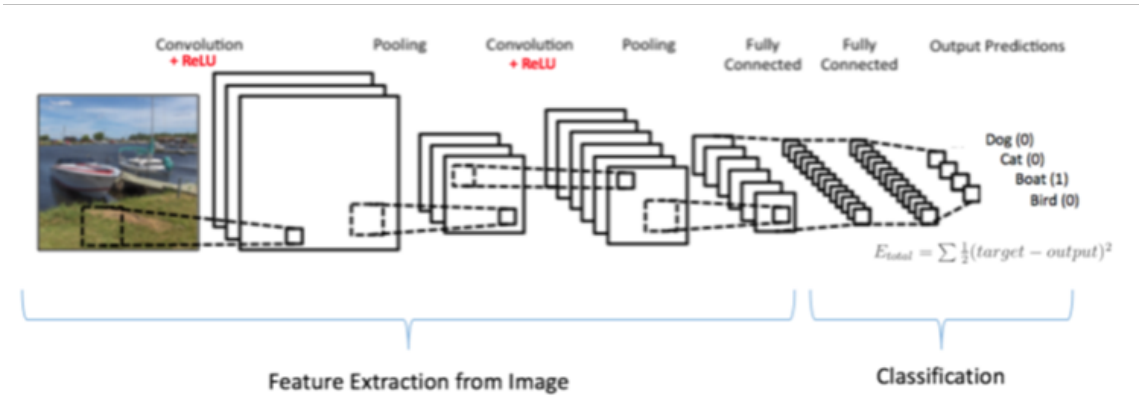


Figure 2.7: Example of a CNN, classifying images into four categories: dog, cat, boat and bird [85].

2.5.1 Convolutions

The one-dimensional convolution operation is defined as

$$s(t) = (x * w)(t) = \int x(a)w(t-a)da, \tag{2.20}$$

where, in the case of CNNs, x would be the referred to as the input, and w as the **kernel** [45]. The output is often referred to as the *feature map* or *activation map*. The discrete convolution is defined as

$$s(t) = (x * w)(t) = \sum_{a=-\infty}^{\infty} x(a)w(t-a). \tag{2.21}$$

Often, in CNNs, convolutions are used over more than one layer at the time, for example for a two-dimensional image, I,

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(m,n)K(i-m, j-n), \tag{2.22}$$

where K is a two-dimensional kernel. Convolution is a commutative operation. An operation $\star$ on a set S, is commutative if

$$a \star b = b \star a \tag{2.23}$$

for every $a, b \in S$ [35]. How this operation is used in CNNs, and its advantages, are described in the following section.

2.5.2 Convolutional Layers

In convolutional layers the activation maps from the previous layers are convolved with a set of small filters (kernels). One can think of each filter as sliding over all spatial locations of the input, computing the *feature map* for each area. The feature maps are then passed through nonlinear activation functions, typically ReLU, to produce the activation maps output [73]. Figure 2.8 illustrates the output of a two-dimensional (2D) convolution.

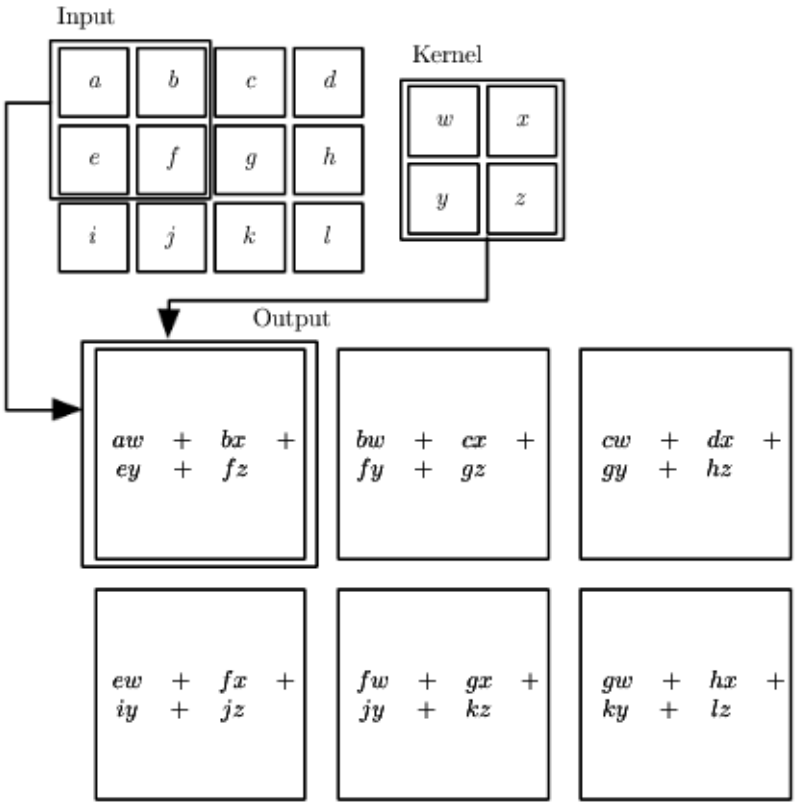
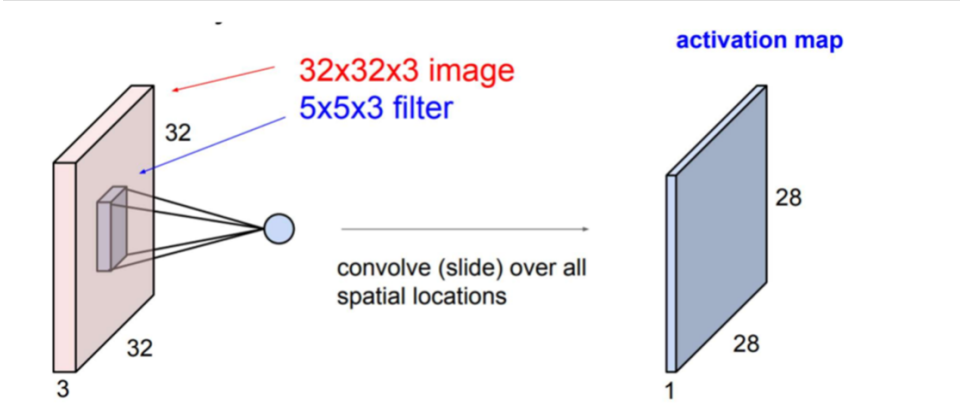
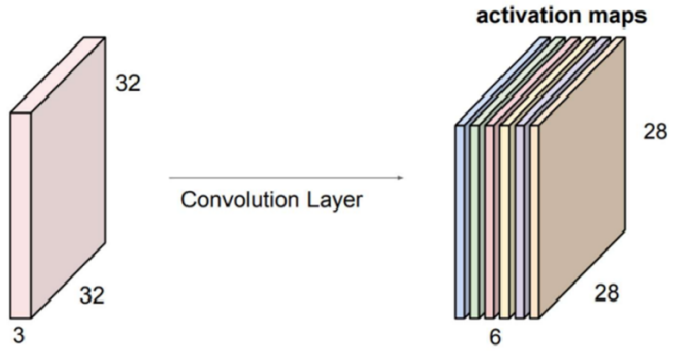


Figure 2.8: Example of a 2D convolution with output restricted to positions where the kernel lies entirely within the image. Input of dimensions 4×3 *with kernel of dimensions* 2×2 *yield output of dimensions* 3×2 *[45].*

The three main advantages of CNNs are *sparse interactions, parameter sharing* and *equivariant representations*. CNNs can also work with input of variable size [45]. Sparse interactions refer to the filters of convolutional layers being smaller than the input. This means having to store fewer parameters, which reduces memory cost and improves efficiency of training, as well as computing output requires fewer operations.



(a) Example of output from a single convolutional filter, or kernel, of dimensions $5 \times 5 \times 3$*, convolve over a* $32 \times 32 \times 3$ *input image, producing a* $28 \times 28 \times 1$ *activation map. The depth of the filters must match the depth of the input.*



(b) Input image of dimension $32 \times 32 \times 3$ *being filtered by six* $5 \times 5 \times 3$ *convolutional filters produces six* $28 \times 28 \times 1$ *activation maps, one for each filter.*

Figure 2.9: Simple illustration of convolutional layer of six $5 \times 5 \times 3$ *filters [85].*

In traditional NNs, a parameter is used only once when computing the output of a layer. In CNNs, each parameter of a filter is used at every position of the input, excepting maybe some boundary pixels. This is referred to as parameter sharing, and means that instead of learning a separate set of parameters for every input location, only one set is learned, which greatly reduces the storage requirements of the model [45].

Equivariant representations refer to that the parameter sharing of CNNs is equivariant to translation. A function *f(x)* is equivariant to a function g if $f(g(x)) = g(f(x))$ [45]. In practice, this means that the output is the same whether we apply a transformation to the image first, then the convolution, or the convolution first, then the transformation. For example, if we move the object in the input, its representation will move the same in the output.

Simply put, one filter can be thought of as learning to detect one single feature from the input, e.g. edges or horizontal lines, which is useful for the whole input image, not just one location.

2.5.3 Pooling Layers

A *pooling function* replaces the output of the net at a certain location with a summary statistic of nearby inputs [45]. Among the most common pooling functions are *max pooling* and *average pooling*. Max pooling returns the maximum value within a rectangular neighbourhood, while average pooling returns the average. Pooling layers performs downsampling of the input, i.e. reduces the size, and makes the representation *invariant* to small translations of the input. This means that if the input is translated by a small amount, the pooled outputs do not change (by much), which is useful if we are only interested in whether a feature is present, rather than where it is. Figure 2.10 illustrates max pooling with 2×2 filters.

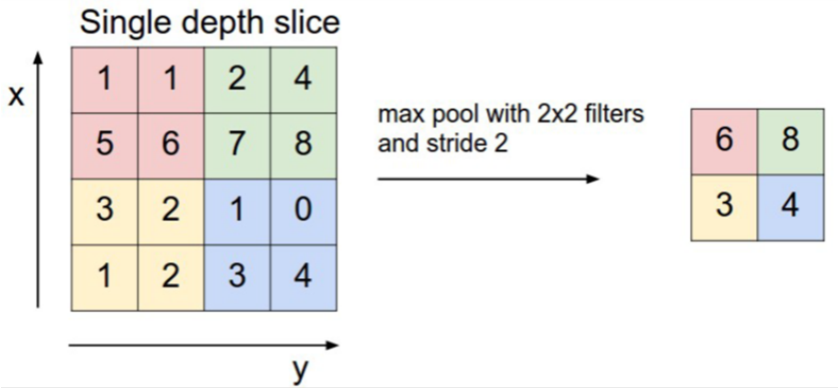


Figure 2.10: Example of max pooling with 2×2 filters. ***Stride*** *refer to how many locations (along the x or y axis) to shift before computing new output [85].*

2.5.4 DenseNet

There exists many CNN architectures, for example *LeNet*, *ResNet*, *GoogleNet/Inception* and *AlexNet* [50; 68; 70; 113]. One CNN architecture, which will be described here, is *DenseNet*. This short description is based on the article "Densely Connected Convolutional Networks" by Huang et al, presented in 2017 [53].

DenseNet, or *Densely Connected Convolutional Networks*, makes use of so-called *dense blocks*, where all layers with matching feature-map sizes are connected. Following are explanations of some key concepts to understand DenseNet, where x_0 denotes a single image passed through a CNN of L layers, each implementing a non-linear transformation $H_l(\cdot)$, where l indexes the layer. In DenseNet, $H_l(\cdot)$ is a composite function of BN, followed by ReLU and a 3×3 convolution. In traditional feed-forward CNNs the output of the l^{th} layer is the input of the $(l+1)^{th}$ layer, such that the layer transition can be given as

$$x_l = H_l(x_{l-1}). \tag{2.24}$$

Dense blocks. In dense blocks, direct connections from any layer to all subsequent layers are set to promote information flow between layers. Let $[x_0, x_1, ..., x_{l-1}]$ be the concatenation of the feature-maps produced in layers 0, ..., l-1, then the layer transition

Layers	Output Size	DenseNet-121	DenseNet-169	DenseNet-201	DenseNet-264
Convolution	112 × 112	7 × 7 conv, stride 2			
Pooling	56 × 56	3 × 3 max pool, stride 2			
Dense Block (1)	56 × 56	[1 × 1 conv; 3 × 3 conv] × 6	[1 × 1 conv; 3 × 3 conv] × 6	[1 × 1 conv; 3 × 3 conv] × 6	[1 × 1 conv; 3 × 3 conv] × 6
Transition Layer (1)	56 × 56	1 × 1 conv			
	28 × 28	2 × 2 average pool, stride 2			
Dense Block (2)	28 × 28	[1 × 1 conv; 3 × 3 conv] × 12	[1 × 1 conv; 3 × 3 conv] × 12	[1 × 1 conv; 3 × 3 conv] × 12	[1 × 1 conv; 3 × 3 conv] × 12
Transition Layer (2)	28 × 28	1 × 1 conv			
	14 × 14	2 × 2 average pool, stride 2			
Dense Block (3)	14 × 14	[1 × 1 conv; 3 × 3 conv] × 24	[1 × 1 conv; 3 × 3 conv] × 32	[1 × 1 conv; 3 × 3 conv] × 48	[1 × 1 conv; 3 × 3 conv] × 64
Transition Layer (3)	14 × 14	1 × 1 conv			
	7 × 7	2 × 2 average pool, stride 2			
Dense Block (4)	7 × 7	[1 × 1 conv; 3 × 3 conv] × 16	[1 × 1 conv; 3 × 3 conv] × 32	[1 × 1 conv; 3 × 3 conv] × 32	[1 × 1 conv; 3 × 3 conv] × 48
Classification Layer	1 × 1	7 × 7 global average pool			
		1000D fully-connected, softmax			

Table 2.1: DenseNet architectures. The growth rate for all the networks is k=32. Each "conv" layer in the table corresponds to the BN-ReLU-Conv sequence described [53].

in dense blocks is given by

$$x_l = H_l([x_0, x_1, ..., x_{l-1}]). \tag{2.25}$$

Bottleneck Layers. To reduce the number of input feature-maps, a 1×1 convolution *bottleneck layer* is implemented before each 3×3 convolution. In these layers the $H_l(\cdot)$ transformation consists of BN, ReLU and a 1×1 convolution.

Transition Layers. Deep DenseNets consists of multiple dense blocks, between the dense blocks, in order to provide down-sampling in the network, are *transition layers*. These layers do convolution and pooling. More particularly, they consists of BN, 1×1 convolution and 2×2 average pooling.

Growth rate. Let k be the number of feature maps produced by H_l, then the l^{th} layer has $k_0 + k \times (l-1)$ input feature maps where k_0 is the number of channels in the input layer. Hyper-parameter k is known as the *growth rate* of the network.

Compression. The hyper-parameter $\theta \in (0, 1]$, known as the *compression factor*, further improves model compactness by reducing the number of feature maps at transition layers. Let a dense block contain m feature maps, then the following transition layer generates $[\theta m]$ output feature maps. $\theta = 1$ is the same as no compression.

The DenseNet-121, DenseNet-169, DenseNet-201 and DenseNet-264 architectures are given in table 2.1. These architectures consists of four dense blocks and the number suffix refer to the deepness of the dense blocks. The information flow of DenseNet help alleviate the problem of vanishing gradients in deep networks, while still being relatively efficient to train in spite of its deepness, as the denseness of the network requires fewer parameters than traditional CNNs.

2.5.5 Occlusion Sensitivity

While DL models enable superior performance in many task areas [53; 73; 90; 104; 125], they are difficult to interpret. This lack of interpretability is generally an issue as knowing how the model came to its conclusions is useful in itself, but becomes striking when the models fail miserably, and one has no idea of why. Therefore, good visualization of DL model decisions is an area of research in itself [60; 104; 125]. *Occlusion sensitivity* is one such visualization tool for understanding how a model makes its predictions.

In occlusion sensitivity, the difference in the prediction score is measured when part of the input is masked, or in other words *occluded* [104]. For images, this means setting an occlusion mask, and slide it over all locations of the image, measuring the difference in prediction score for all locations. The results are then typically illustrated with heatmaps. "Cold" areas of these heatmaps will then indicate important areas of the image for making the prediction, as occluding these areas led to drop in the prediction probability score. Figure 2.11 show occlusion maps with different mask sizes for a CNN in predicting "tiger cat" from one of the ablation studies of the article "Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization" by Selvaraju et al. [104]. The blue regions indicates evidence for the the class, while the red regions indicate regions that led to an increase in prediction score, i.e. areas that confuses the model in its prediction, possibly giving evidence of another class.

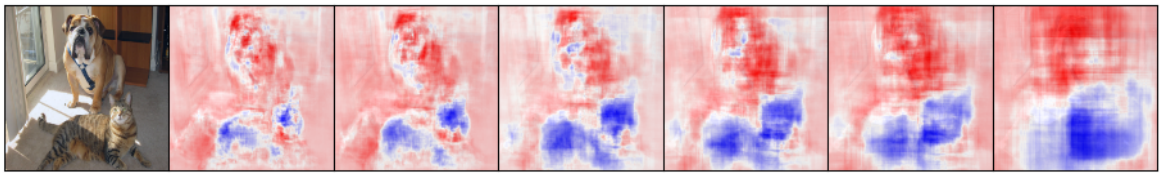


Figure 2.11: Occlusion maps with different mask sizes for CNN predicting "tiger cat" category.

2.6 Alzheimer's Disease and Magnetic Resonance Imaging

2.6.1 The Human Brain

The human brain is the central organ of the human nervous system. It consists of the *cerebrum*, the *cerebellum* and the *brain stem*. Main regions of the brain and some of their functions is shown in figure 2.12.

2.6.1.1 Cerebrum

The cerebrum is the largest region of the human brain. The surface of this region is made up of elevated ridges of tissue, called *gyri* that are separated by shallow groves called *sulci* [74]. The larger regions are separated by deeper groves, called *fissures*. The tissue of this part of the brain is separated into gray and white matter. The outer cortex of the cerebrum, the *cerebral cortex*, is composed of grey matter, while the inner region is composed of white matter, as well as there being some islands of grey matter

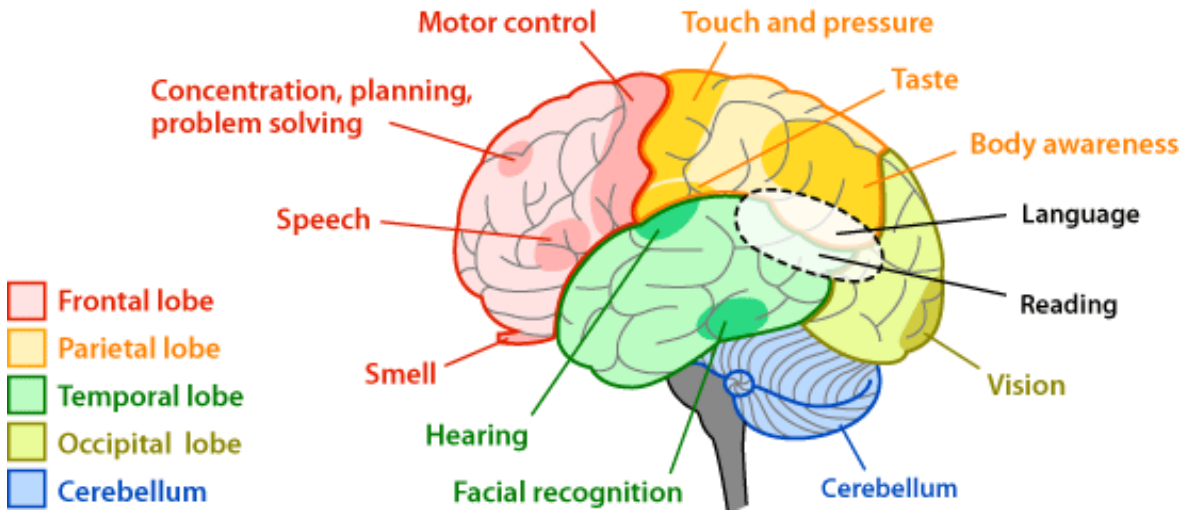


Figure 2.12: Regions of the human brain and location of some brain functions [114].

deep within the white matter, called the *basal nuclei* [74]. The cerebral cortex is where our conscious mind is found [74]. The cerebrum is divided into two hemispheres, left and right, by a deep fissure called the *longitudinal fissure*. These two hemispheres communicates mainly through the *corpus callosum* [81]. Each hemisphere controls the movements of the opposite side of the body, i.e. the left side of the brain controls muscles of the right side of the body, and vice versa. The cerebrum is divided into four different lobes: the *frontal lobe*, the *parietal lobe*, the *temporal lobe* and the *occipital lobe*. Though all lobes work together, each is associated with specific functions [81]. The names of gyri, sulci and other areas of the different lobes, are illustrated in figure 2.13.

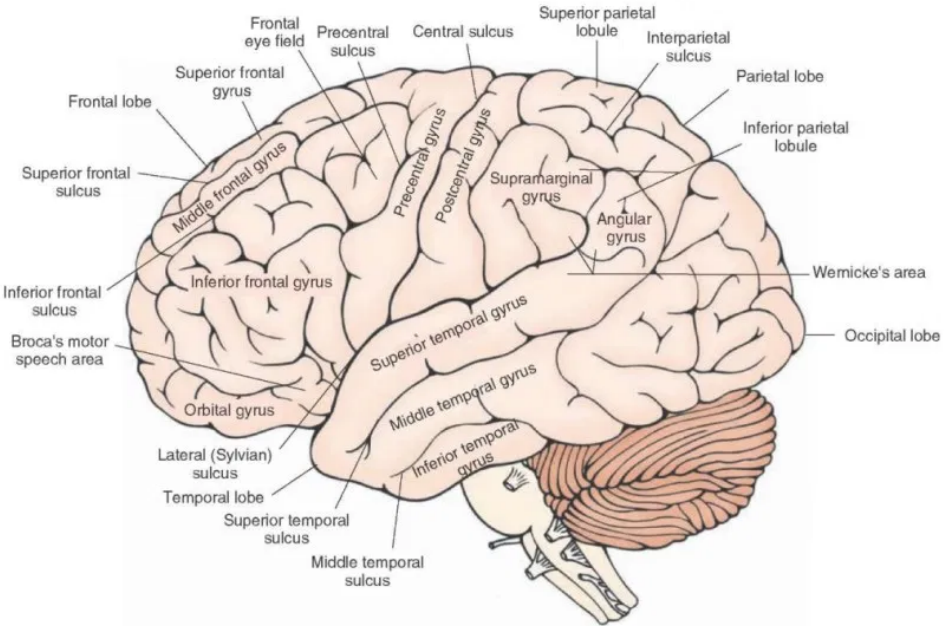


Figure 2.13: The human brain anatomy [94].

The Deep Structures of the Brain

The *diencephalon* forms the central core of the brain, surrounded by the cerebral hemispheres [74]. It mainly consists of the *thalamus, hypothalamus and epithalamus*, and plays a vital role in integrating sensory information and motor commands. The *caudate nucleus, putamen and globus pallidus*, important regulators of skeletal muscle movement strongly connected to the thalamus, make up the basal nuclei [74]. The *limbic system* consists of structures at the edges of the left and right hemispheres of the cerebrum, in the "middle" of the brain, encircling the upper part of the brain stem. These structures include the *amygdala, hippocampus, cingulate gyrus, parahippocampal gyrus,* and the *hypothalamus* and the *anterior thalamic nuclei* in the diencephalon [74]. The limbic system is associated with memory, learning, emotions and subconscious body functions. The *venticles* are hollow chambers in the brain, continuous with one another and the central canal of the spinal cord [74]. These chambers are filled with *cerebrospinal fluid*, which functions as a cushion and nourishment for the brain.

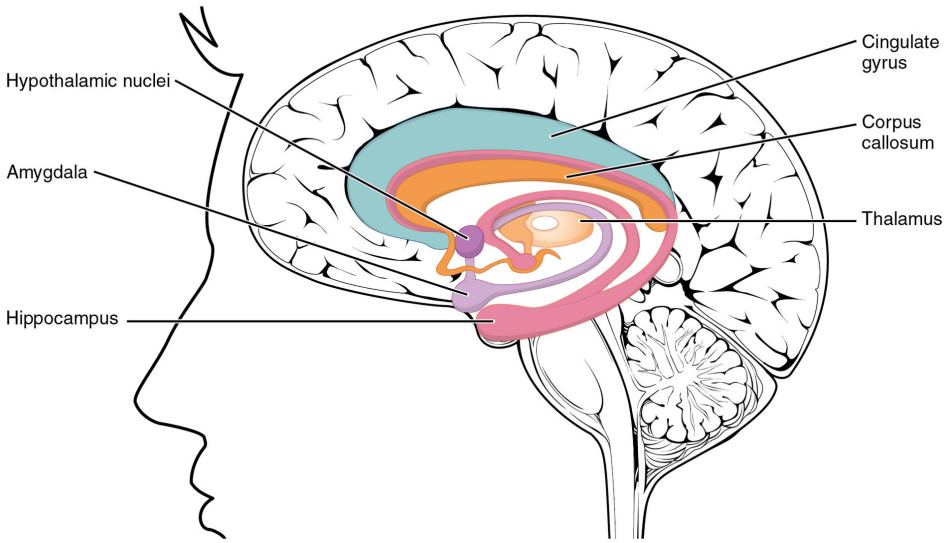


Figure 2.14: The limbic lobe and surrounding structures [19].

2.6.1.2 Cerebellum

Like the cerebrum, the cerebellum is divided into two hemispheres, the outer region is made up of gray matter, and the inner region of white matter. It is separated from the cerebrum by the *transverse cerebral fissure* [74]. The cerebellum is associated with coordinating movements [81].

2.6.1.3 Brain Stem

The brain stem is the brain's connection to the spinal cord. Three main structures make up the brain stem: the *midbrain, pons* and *medulla oblongata*. The latter is associated with vital functions such as heart beat, breathing, swallowing and blood pressure, while the pons is involved with coordinating eye and facial movements, hearing and balance, and the midbrain with eye movements [11]. The brain stem is also associated with sleep and coordinating many important reflexes [81].

2.6.2 Magnetic Resonance Imaging

The basis for *magnetic resonance imaging* (MRI) is a chemical analytical technique called *nuclear magnetic resonance imaging* (NMR) [48]. MRI utilizes the signal of reflected radio frequency waves from magnetized spins in the body of the patient. The patient is placed in an external magnetic field, which will cause the spins of its protons to line up with the field [48]. A radio frequency wave of a specific frequency is then sent into the patient, which will cause some spins to change their alignment. Then they reflect a signal, as they return to their previous alignment, and it is this signal which is measured. To differentiate where, from the body, the signal is coming from, a technique called *spacial encoding* is used. The spacial encoding utilizes three orthogonal gradient coils, corresponding to the axes x, y and z in a three-dimensional coordinate system [48]. A coil is an electrical device composed of multiple loops of wire. Gradient coils generate magnetic fields, and can intentionally cause modifications to the uniformity of the magnetic field. These modifications or *perturbations* can be used for deciphering spatial information of the received signal.

MRI can be either *T1* or *T2 weighted*. T1 and T2 refer to *relaxation times*; i.e. the timing of the radio frequency pulse intervals. The term relaxation comes from the time it takes for the spins to relax back to the equilibrium state [49]. T1 refers to the time it takes for the spins to realign along the longitudinal (z)-axis. T2 refers to the time it takes for the spins to get out of phase with one another in the x-y plane. Practically, T1 weighted images highlight fat tissue, while T2 weighted images highlight both fat tissue and water.

2.6.3 Alzheimer's Disease

Alzheimer's Disease (AD) is the most common cause of dementia, which is a general term for decline in cognitive abilities severe enough interfere with daily life, among the elderly. AD is marked with synaptic loss, shrinking brain volume, death of neurons, neuro-inflammation and *plaques* and *tangles* in the brain. Synaptic dysfunction is one of the first events in AD [61]. Plaques and tangles was first discovered by Dr. Alois Alzheimer in 1906, in the autopsy of the brain of a patient of his, Miss Augusta "D" [61]. Miss Augusta became the first known patient of AD. Amyloid plaques form as a result of the deposition of amyloid beta peptides (Aβ) originating from the cleavage of amyloid precursor protein (APP) [61]. When the amyloid plaques become big enough, they can block cell to cell communication and the transfer of necessary nutrition for the cells. Neurofibrillary tangles are generally abnormal accumulations of hyperphosphorylated tau proteins inside neurons [38]. In humans, tau is a heat stable protein essential for microtubule assembly, mainly found in neurons [92]. Microtubules are part of the cytoskeleton, important for cellular trafficking, so one direct consequence of neurofibrillary tangles is that the transport of nutrients and other important cargo fails. Amyloid plaques and neurofibrillary tangles cause neuronal death.

Brain atrophy, or brain volume decrement, is part of normal aging, but in AD this process is accelerated. This acceleration is mainly attributed to cell death [57]. The deposition of Aβ in the brain, a biomarker of AD [75], is associated with accelerated brain

atrophy [38]. The first brain areas affected by AD is the medial temporal lobe; starting at the entorhinal cortex, closely followed by hippocampus, amygdala, parahippocampus and other structures of the limbic lobe [57]. Reduction of hippocampus volume is a key biomarker of AD [38]. Hippocampus is associated with short term memory. The losses then spread to the temporal neocortex and then all areas associated to the neocortex [57]. Even in mild AD indivduals, entorhinal volumes are reduced by 20-30% and hippocampal volumes by 15-25% [57].

Mild cognitive impairment (MCI) is a predemetia stage where the people around notices that something is going on with the person's memory or cognitive functions [38]. Only a small portion of people with MCI will progress to AD after 5-10 years, most will not progress to AD even after 10 years of follow-up [57].

2.6.3.1 Visibility of AD on MRI

As MRI can perform measures of brain anatomy and function, the symptoms of AD and other neurological disorders are visible in MRI data [27]. MRI can be used to highlight brain atrophy, and reduction in hippocampus volume is derived from structural MRI [38]. In figure 2.15, MR images of healthy and AD brain are shown. The atrophy of AD is clearly visible. It's best to use T1-weighted volumetric sequences MRI to visualize cerebral atrophy [57].

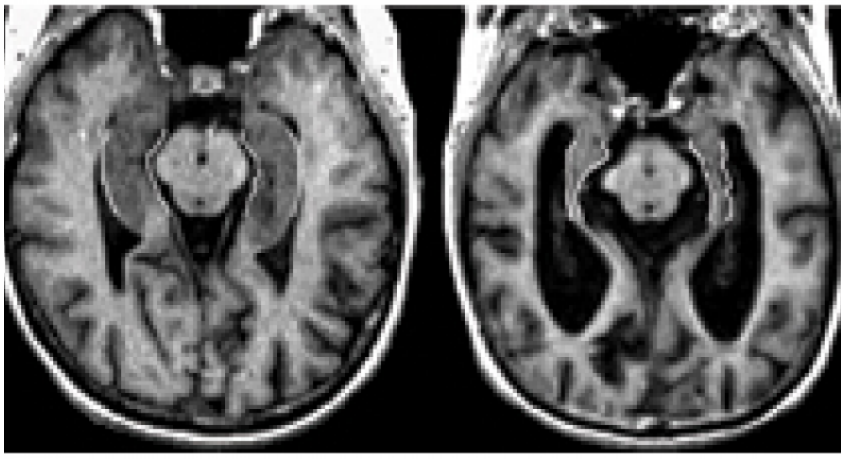


Figure 2.15: MR images of healthy brain (left) and AD brain (right) [10].

2.7 Genetics

2.7.1 Genome-Wide Association Studies and Imaging Genetics

Genome-wide association studies (GWAS) investigates associations between single nucleotide polymorphisms (SNPs) and phenotypes [118]. A SNP is a variation in nucleotide base at a single position in a DNA sequence that are quite common between individuals of the same species [78]. The variation must exist in at least 1% of the population for the nucleotide base variation to be defined as a SNP [117]. SNPs can occur both in genes, in which case it is said that the gene has more than one allele, and in

non-coding parts of DNA.

Imaging genetics studies the complex associations between genetic variables and imaging phenotypes (IPs) [25]. Neuroimaging genetic studies assess the impact of genetic variables, typically SNPs, on brain function [55]. Generally, the aim of these studies are increasing our understanding of how genetic variants impact brain structure and function, which is highly important with regard to diagnosis, prevention and treatment of complex brain-related disorders such as AD [80].

2.7.2 Genetic Risk Factors of AD

GWAS have so far found associations between AD and a vast set of genetic loci, including apolipo-protein E (APOE), phosphatidylinositol binding clathrin assembly protein (PICALM), bridging integrator 1 (BIN1) and clusterin (CLU) genotypes, amongst others [12; 91; 106]. The apolipo-protein E (APOE) has long been known as the strongest genetic risk factor of AD, with carriers of the APOE $\varepsilon 4$ allele having the greatest risk through the reduction of Aβ clearance [12]. In addition to APOE, several AD risk genes, have been found to be associated with Aβ clearance and aggregation, such as PICALM, CLU and complement receptor 1 (CR1). Furthermore, of known AD genetic risk loci, rs2829887 of chromosome 21, located close to the APP gene, rs3764650 of adenosine triphosphate–binding cassette subfamily A member 7 (ABCA7) and rs9349407 of CD2-associated protein (CD2AP), have also shown relation to neurotic plaque burden [107].

Imaging genetics studies have found associations of some AD risk loci with brain neurodegeneration; such as rs3851179 of PICALM, rs7561528 of BIN1, rs1408077 of CR1 and rs3764650 of ABCA7, have been found to be associated with cortical and hippocampal atrophy [12].

Updated data on genes and SNPs associated with AD is readily available on many platforms. Amongst them, the AlzGene Database aim to "(...) serve as a comprehensive, unbiased, publicly available and regularly updated field-synopsis of published genetic association studies performed on AD phenotypes" [18], and keep a ranking of top genetic risk factors of AD[1].

2.7.3 Machine Learning in Imaging Genetics

The earliest methods for imaging genetic studies involve selecting a few candidate regions of interest (ROI) in the brain and specific candidate genetic markers, and perform univariate analysis of association, typically using linear regression. These methods have been extended to brain-wide genome-wide studies by performing massive numbers of pair-wise univariate analyses [80]. Voxel-wise approaches are build on this principle, by fitting separate models to each voxel of the image, but may include multiple genetic markers in each model. In 2010, Stein et al. performed a voxel-wise GWAS examining association of 448,293 SNPs with 31,622 voxels of the entire brain [111].

[1] https://web.archive.org/web/20110222233234/http://www.alzgene.org/TopResults.asp

In 2015, Huang et al. developed a fast voxel-wise GWAS (FVGWAS) approach, aiming to increase the computational efficiency of brain-wide genome-wide studies, and applied it to assess 501,584 SNPs with 193,275 voxels [54].

An alternative to such pair-wise univariate analysis, have been to perform multivariate high-dimensional regression to entire datasets. Such procedures are highly computationally expensive, so the scale of the data must be reduced. Commonly, information of the entire image is summarized using a relatively small number of brain summary measures across some key ROIs [80]. In 2012, Wang et al. developed such approaches in their imaging genetic studies of AD related SNPs [119; 120].

Disadvantages of massive univariate and voxel-wise approaches include that they may be computationally expensive, and that the relationship between the different IPs and between the different genetic markers are lost [80]. Several areas of the human brain are typically involved together in many brain functions, and multiple SNPs from one gene often jointly carries out genetic functions [119]. The multivariate regression approach address the latter problem, however, the selection of IPs to make it computationally feasible, takes time, and requires knowledge of which features is wise to choose.

This study will use machine learning approaches in order to select appropriate features, either in the form of feature selection of manually extracted features from MRI data, or through the power of DL models in finding important structures themselves. DL is becoming a methodology of choice for analyzing medical images, be it for image classification, object detection, segmentation, registration and other tasks [105]. The image features automatically learned by a CNN during training, have long surpassed what can be engineered by hand [73], and CNNs have therefore seen many applications within the field of MRI; image reconstruction, image restoration, image segmentation and disease prediction to name a few. As an example, many studies have successfully used CNNs in predictions of AD and/or MCI status on MRI data [14; 71; 98; 115; 121]. In 2018, Lin et al. trained a CNN to predict MCI subjects that will convert to AD on extracted features from MRI data, that achieved a leave-one-out cross validation accuracy of 79.9% [71]. The same year, Tang et al. performed both binary and 3-way classifications of AD diagnosis directly on 3D MRI using a 3D CNN inspired by the VGG architecture, with the 3-way classification of AD/MCI/CN achieving an accuracy of 91.32% [115]. Another approach of classification of AD directly on 3D MRI, yielding good results, have been ensembles of 3D DenseNets [98; 121].

2.8 Statistics

2.8.1 *P*-value Hypothesis Testing

In scientific research, the null hypothesis is the default hypothesis that chance alone is responsible for any observed results, there is no relationship or effect between the data [51]. It is the assumed true hypothesis that a researcher tries to disprove; the opposite of the alternative hypothesis, which is what the researcher thinks is the cause of the phenomenon. The p-value is the probability of obtaining a result better than or equal

to the observed result, under the assumption that the null hypothesis is true [15]. If the p-value is smaller than a threshold value decided on beforehand, the null hypothesis is rejected. A common threshold to choose is 0.05.

2.8.2 Permutation Tests

A common tool for p-value hypothesis testing is permutation tests. During these tests, the labels of the dataset are rearranged, and the performance of a chosen statistical model is measured with some statistic. The performance of the models with permuted data are then compared with the test statistic, which is the performance of the model with the original data. From this, it follows that the null hypothesis is simply that the labels of the dataset are interchangeable [63]. The p-value is then the probability of obtaining a result better than or equal to the test statistic when the data is permuted.

As an (absurd) example of a hypothesis, H, let us say we look at the connection between getting a haircut and developing a lung infection. Our alternative hypothesis ($H = 1$) is that getting a haircut is a possible cause for lung infection, while the null hypothesis ($H = 0$) is that it is not. Our dataset consists of data on whether the subject has recently got a haircut, and whether the subject has lung infection. Then, we choose an appropriate ML classification model to investigate this. The test statistic, t, is thus the performance of the model on the data. The p-value is the probability of getting a result better than or equal to the test statistic, given that getting a haircut does not affect developing lung infection. This can be written as the conditional expression:

$$p-\text{value} = \text{P}(T \geq t | H = 0) \tag{2.26}$$

where T is the *permuted value*, the performance of the model on permuted data. The threshold, α, for our p-values is set to 0.05, a commonly chosen threshold. This means that we reject he hypothesis that getting a haircut is not a cause of lung infection if the model's performance on permuted data is better than on the original data in less than 5% of all permutations. Essentially, this is the same as saying we allow for a 5% chance of being wrong when rejecting the null hypothesis.

Theoretically, the p-value should be assessed by computing the performance of all possible permutations of the data. However, this is generally not feasible in practice. Given a binary dataset with equal distribution of the classes, for ten entries the number of possible permutations is $\binom{10}{5} = 252$, for 100 entries the number is $\binom{100}{50} \propto 10^{29}$. Therefore, it is common to approximate the p-value by a limited number of permutations [63]. The p-value is usually calculated as the number of times the performance on the permuted data is equal or surpasses the test statistic divided by the total number of permutations. A pseudo-count of one is usually added to avoid p-values of zero [63]. Mathematically,

$$p-\text{value} = \frac{M+1}{N+1} \tag{2.27}$$

where M is the number of times $T \geq t$ and N is the total number of permutations. Thus p-value$\in [\frac{1}{N+1}, 1.00]$ where $p-\text{value} = \frac{1}{N+1}$ is the best and $p-\text{value} = 1.00$ is the worst.

2.8.2.1 Permutation Tests as a Way of Estimating Classifier Performance

Estimating error or accuracy with cross validation are amongst the most common methods for measuring performance of classifiers. However, such performance measures are considered insufficient when the dimensionality of the data is too high, there are too few datapoints or the data is significantly imbalanced [16; 82]. Permutation tests are useful in assessing whether a classifier has found a real class structure in the data [82]. Given a binary classification model on a dataset with a hundred features, fifty datapoints, a majority class represented by 75% of the datapoints, and only a cross-validation accuracy of 0.74 as performance measure, there is no way of knowing from whether the model has learned anything from the data except the class distribution. However, say it has a p-value of 0.007. This means that the model performed better on permuted data only 0.7% of the times, making it significantly likely that the model has learned some structure other than that.

2.8.3 Multiple Hypothesis Testing

When performing multiple null hypothesis tests, the false positive rate, i.e. number of times the null hypothesis is falsely rejected, will increase with the number of tests [17]. As GWAS involve null hypothesis testing for large numbers of genetic loci spanning a large portion of the genome, approaches to measuring statistical significance of the results are necessary [112].

2.8.3.1 False Discovery Rate

Following will be an explanation of the approach to control for the False Discovery Rate (FDR), presented by Benjamini and Hochberg in 1995 [17]. FDR is defined as:

$$Q_e = E(\mathbf{Q}) = E\{\mathbf{V}/(\mathbf{V}+\mathbf{S}\} = E(\mathbf{V}/\mathbf{R}), \tag{2.28}$$

where $\boldsymbol{V}$ is the number of true null hypothesis declared significant (false discoveries), $\boldsymbol{S}$ is the number of false null hypothesis declared significant (true discoveries), $\mathbf{R} = \mathbf{V}+\mathbf{S}$, and $\boldsymbol{Q}$ is the proportion of the rejected null hypothesis which are wrongly rejected: $\mathbf{Q} = \mathbf{V}/(\mathbf{V}+\mathbf{S})$. In other words, FDR, or Q_e, is the expectation of $\boldsymbol{Q}$.

Let $P_1, P_2, ..., P_m$ be the corresponding p-values to null hypothesis tests $H_1, H_2, ..., H_m$. Let $P_{(1)} \leq P_{(2)} \leq ... \leq P_{(m)}$ be the ordered p-values, and denote $H_{(i)}$ as the null hypothesis corresponding to $P_{(i)}$. The *Benjamini-Hochberg procedure* is then defined as:

$$\begin{aligned} &\text{let } k \text{ be the largest } i \text{ for which } P_{(i)} \leq \frac{i}{m}q^*, \\ &\text{then reject all } H_{(i)}\ i = 1, 2, ..., k. \end{aligned} \tag{2.29}$$

For independent test statistics and for any configuration of false null hypothesis, the above procedure controls the FDR at q^*.

3 Methods

3.1 Materials

All data used in the preparation of this thesis were obtained from the ADNI database[1]. The ADNI was launched in 2003 as a public-private partnership, led by Principal Investigator Michael W. Weiner, MD. The primary goal of ADNI has been to test whether serial MRI, positron emission tomography (PET), other biological markers, and clinical and neuropsychological assessment can be combined to measure the progression of mild cognitive impairment (MCI) and early Alzheimer's disease (AD). For up-to-date information, see www.adni-info.org. For this study, both raw and processed MRI data, genotype data, diagnostic data as well as data on gender and age were used.

3.1.1 Study Population

The data in the ADNI database are collected from elderly individuals, the youngest being in their fifties, and the mean being in the mid-seventies. For the RF classification tasks, hereby termed "phase 1", the cohort included all ADNI1 participants that are part of the ADNI1:Complete 1Yr 1.5T collection. This included 628 individuals, where 190 were normal controls (CN), 305 were diagnosed with mild cognitive impairment (MCI) and 133 with early AD [88]. For the deep learning classification tasks, hereby referred to as "phase 2", the cohort included MRI data from 1619 ADNI2 participants, this includes the participants of ADNI1 and ADNI GO. Of these, there were 387 CN, 285 diagnosed with early MCI (EMCI), 532 diagnosed with late MCI (LMCI), 315 diagnosed with AD, and 100 belonging to another group termed significant memory concerns (SMC). This latter group consist of subjects manifesting more subtle degrees of cognitive manifestations either experienced by themselves or observed by others.

[1] adni.loni.usc.edu

Table 3.1: Distribution of the study population into the diagnostic study groups for the data used for the RF and DL classification tasks.

Study Group	**RFC**	**DL**
CN	190	387
LMCI	305	532
EMCI	-	285
AD	133	315
SMC	-	100
Total	628	1619

The MCI participants from former phases have been included in the LMCI group. All of these groups are diagnosis at baseline. See table 3.1 for overview of the diagnostic groups in the two phases. Of the cohort for phase 1, 42.0% were female, of the latter cohort, 44.7% were female. Another point of interest is the distribution of the carrier status of *APOE* ε4. Of the phase 1 subjects, 51.4% carried no *APOE* ε4 alleles (group APOE4=0), 37.3% carried one *APOE* ε4 allele (group APOE4=1), and 11.3% carried two *APOE* ε4 alleles (group APOE4=2). Of the phase 2 dataset 53.2% belonged to group APOE4=0, 36.9% to group APOE4=1 and 9.9% to group APOE4=2. The distributions of gender and *APOE* ε4 carrier status per diagnostic group are visualized in figure 3.1.

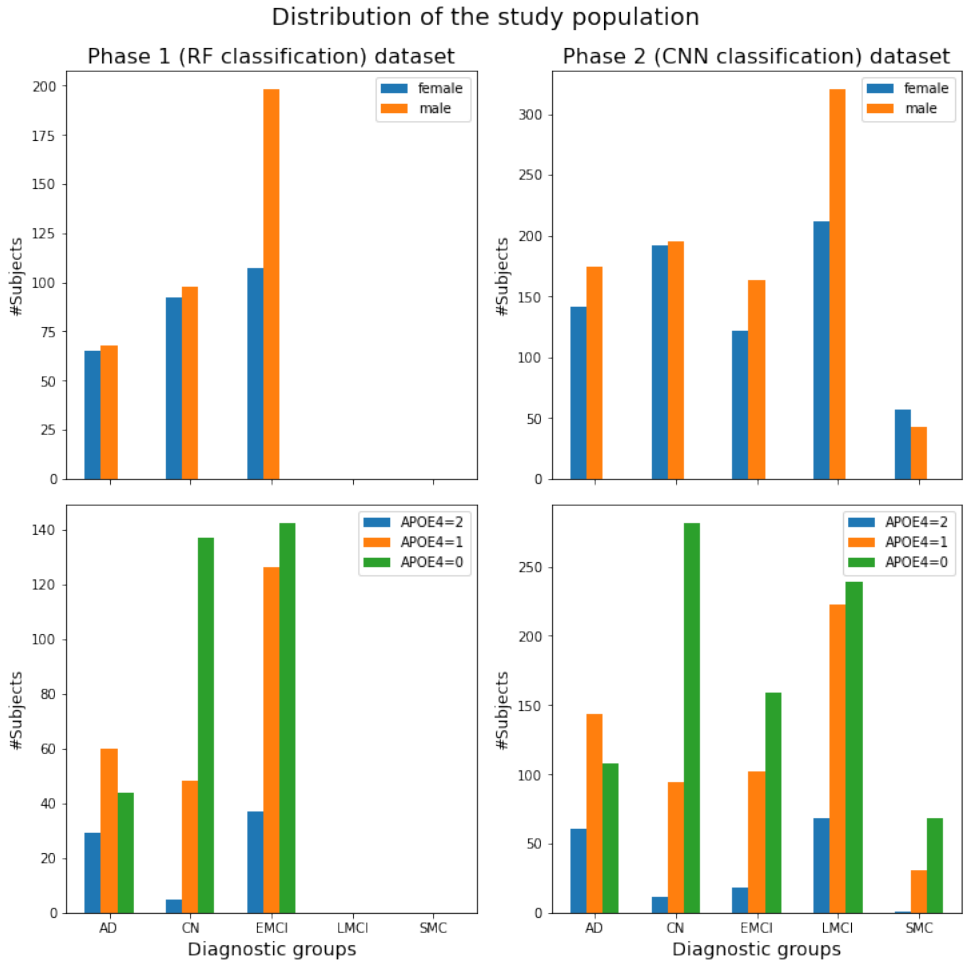


Figure 3.1: Distributions of gender and APOE4 over the diagnostic groups for the two datasets used in phase 1 and phase 2, respectively

3.1.2 MRI Data Preprocessing for Phase 1: the RFC Tasks

A big part of this study have been RF classification on manually extracted features from MRI data. Typically, these features are extracted by segmenting the brain structure into anatomical regions or tissue types, and then calculating different characteristics such as region volumes and tissue thickness [90]. For this, the image file dataset accessed

at the ADNI database at Download > Study Data > Test Data > "AD Challenge Training Data:Imaging", were used. This dataset contains exactly such features. The source data consists of 1.5 Tesla T1-weighted MRI volumes from the baseline scans of the ADNI1:Complete 1Yr 1.5T Data Collection. These source images are in NiFTI (.nii) format, and have been through the following preprocessing by Mayo Clinic: gradient warping, scaling, B1 correction and N3 inhomogeneity correction. To extract features, the images have been processed using the neuroimaging software pipelines: FreeSurfer, Advanced Normalization Tools (ANTs) and Mindboggle [88]. Following is the description and correct citation of the FreeSurfer pipeline for the dataset as described at http://surfer.nmr.mgh.harvard.edu/fswiki/FreeSurferMethodsCitation.

> *"Cortical reconstruction and volumetric segmentation was performed with the Freesurfer image analysis suite, which is documented and freely available for download online (http://surfer.nmr.mgh.harvard.edu/). The technical details of these procedures are described in prior publications [21; 22; 28–34; 47; 58; 95; 96; 102]. Briefly, this processing includes motion correction and averaging [95] of multiple volumetric T1 weighted images (when more than one is available), removal of non-brain tissue using a hybrid watershed/surface deformation procedure [102], automated Talairach transformation, segmentation of the subcortical white matter and deep gray matter volumetric structures (including hippocampus, amygdala, caudate, putamen, ventricles)[32; 33], intensity normalization [109], tessellation of the gray matter white matter boundary, automated topology correction [31; 103], and surface deformation following intensity gradients to optimally place the gray/white and gray/cerebrospinal fluid borders at the location where the greatest shift in intensity defines the transition to the other tissue class [21; 22; 28]. Once the cortical models are complete, a number of deformable procedures can be performed for further data processing and analysis including surface inflation [29], registration to a spherical atlas which is based on individual cortical folding patterns to match cortical geometry across subjects [30], parcellation of the cerebral cortex into units with respect to gyral and sulcal structure [23; 34], and creation of a variety of surface based data including maps of curvature and sulcal depth. This method uses both intensity and continuity information from the entire three dimensional MR volume in segmentation and deformation procedures to produce representations of cortical thickness, calculated as the closest distance from the gray/white boundary to the gray/CSF boundary at each vertex on the tessellated surface [28]. The maps are created using spatial intensity gradients across tissue classes and are therefore not simply reliant on absolute signal intensity. The maps produced are not restricted to the voxel resolution of the original data thus are capable of detecting submillimeter differences between groups. Procedures for the measurement of cortical thickness have been validated against histological analysis [97] and manual measurements [69; 99]. Freesurfer morphometric procedures have been demonstrated to show good test-retest reliability across scanner manufacturers and across field strengths [47; 96].*

Aseg Atlas Information

The aseg atlas is built from 40 subjects acquired using the same mp-rage sequence (by people at Wash U ages ago in collaboration with Randy Buckner). The subjects that make up the atlas are distributed in 4 groups of 10 subjects each: (1) young, (2) middle aged, (3) healthy older adults, (4) older adults with AD.

Longitudinal Processing
To extract reliable volume and thickness estimates, images where automatically processed with the longitudinal stream [96] in FreeSurfer. Specifically an unbiased within-subject template space and image is created using robust, inverse consistent registration [95]. Several processing steps, such as skull stripping, Talairach transforms, atlas registration as well as spherical surface maps and parcellations are then initialized with common information from the within-subject template, significantly increasing reliability and statistical power [96]."

Brain volume extraction, segmentation and registration-based labeling were provided by ANTs. Output from FreeSurfer and ANTs were processed with Mindboggle to compute volumes of all labeled regions, thicknesses of all labeled cortical regions and statistical summaries of the shape measures (surface area, travel depth, geodesic depth, mean curvature, FreeSurfer convexity and FreeSurfer thickness) per cortical surface label [88].

3.1.3 MRI Data Preprocessing for Phase 2: the DL Tasks

The MR images used in this phase were skull-stripped and provided by Alexander Selvikvåg Lundervold[2][3] using FreeSurfer version 6.0 'recon-all'. On beforehand, from the ADNI database, the images had the same preprocessing by Mayo Clinic (gradient warping, scaling B1 correction and N3 inhomogeneity correction) as the MR image data used in phase 1, so input to FreeSurfer was T1 weighted images in NiFTI (.nii) format. FreeSurfer 'recon-all' performs all of FreeSurfer cortical reconstruction process [36]. The skull-stripped images used were the output from stage 5 of the process, the Skull Strip stage. Stages 1 to 4 include Motion Correction and Conform, Non-Uniform intensity normalization (NU), Talairach transform computation and Intensity Normalization 1. Description of these stages can be found at https://surfer.nmr.mgh.harvard.edu/fswiki/recon-all/, in short:

1. **Motion Correction** : Corrects for small motions between multiple source volumes and then average them together.
2. **NU Intensity Correction** : Corrects for intensity non-uniformity.
3. **Talairach** : Computes the affine transform from the original volume to the MNI305 atlas using the MINC program mritotal through a FreeSurfer script called talairach.

[2]Associate Professor, Department of Computing, Mathematics and Physics, Western Norway University of Applied Sciences

[3]Senior Data Scientist, Mohn Medical Imaging and Visualization Centre (MMIV), Haukeland University Hospital, Norway

4. **Normalization** : Performs intensity normalization of the original volume.
5. **Skull Strip** : Removes the skull.

The FreeSurfer processing also includes resampling to 1x1x1 isotropic voxels and 256x256x256 dimensions. Outputs from FreeSurfer are in MGZ (.mgz) format, the images were converted to nii.gz (compressed NiFTI) format using 'mri_convert' from FreeSurfer.

3.1.4 Genetic Data

As *APOE* ε4 is the strongest known risk factor for AD [108], ADNI participants were genotyped of *APOE* at enrollment. The *APOE* genotyping was done by extracting DNA from a 3 mL aliquot of EDTA blood using Cogenics [9]. Apart from the *APOE* genotype data, the ADNI1 GWAS (PLINK) dataset, containing genotype information for 620,901 SNPs and CNV Markers, respectively, were used in preparation of this thesis. ADNI1 GWAS was genotyped with Illumina Human610-Quad BeadChip [9].

3.2 Python Modules

All experiments have been performed in the programming language Python. Following is a description of the main modules/frameworks used.

3.2.1 Scikit-learn

Scikit-learn is an open source Python module built on NumPy, SciPy, matplotlib and Cython which provides implementations for machine learning algorithms [87]. A table of all methods and classes used in preparation of this thesis from scikit-learn can be found in Appendix 1 table A.1.1. Their default parameters from the version used are also listed (different versions of scikit-learn have been used in the experiments, that default parameters have not changed through the experimentation have been checked). Documentation can be found at https://sklearn.org/. If not otherwise specified, it can be assumed that these are the methods/classes used for their respective tasks with default settings.

3.2.2 ELI5

ELI5 is a Python library that provides support for scikit-learn amongst other several ML frameworks and packages [67]. The package provides tools for explaining the behaviour of ML models. The class PermutationImportance from ELI5 has been used for the experiments. For documentation see https://eli5.readthedocs.io.

3.2.3 Statsmodels

Statsmodels is a Python module providing implementations, functionality and tools for statistical models, statistical testing and statistical data exploration [101]. The method statsmodels.formula.api.ols from this module has been used to create regression models. See https://www.statsmodels.org/dev/api.html for documentation.

3.2.4 PyTorch

PyTorch is an end-to-end deep learning framework [86]. The deep learning experiments of this thesis have been build with PyTorch.

3.2.5 MONAI

MONAI is a PyTorch-based framework for deep learning in medical imaging [93]. MONAI provides tools for preprocessing of multi-dimensinal imaging data and implementations of deep learning architectures for both 2D and 3D imaging data. A list of classes and methods used can be found in table A.1.2 of Appendix 1. For documentation, see https://docs.monai.io/.

3.3 Phase 1: Random Forest Classifier in Imaging Genetics

The main goal of this phase was to run RF classification of a large number of SNPs (a genome-wide association study), trying to find SNPs associated with AD. Therefore, finding efficient ways of calculating p-values that gave valid results have been a goal of the experiments.

Note: Unless otherwise specified, 42 was used as seed for random state for all methods that requires seeding for reproducible behaviour.

3.3.1 Feature Vectors

From the image dataset described in section 3.1.2, several feature vectors for each of the 628 subjects were extracted, these are described in table 3.2. The following formula was used to normalize the data marked as normalized:

$$\frac{x_i - x_{max}}{x_{max} - x_{min}}, \qquad i \in [0, n) \tag{3.1}$$

where x_i is the i^{th} datapoint of a feature, n is the total number of datapoints, x_{max} is the maximum value of the feature and x_{min} is the minimum value.

The dataset have the following layout:

- For each subject:
  - FreeSurfer volume datas for 243 brain labels
  - FreeSurfer volume and thickness datas for 70 brain labels.
  - For both left and right brain surfaces (separate datasets for each):
    - "Label shapes" dataset, the Mindboggle features (see section 3.1.2) as well as mean position and Laplace-Beltrami spectra for 51 brain labels. The statistical summaries were given in median, MAD, mean, SD, skew, kurtosis, 25% and 75%. .
    - "Sulcus shapes" dataset, same Mindboggle features as above, for 26 brain sulcus.

Table 3.2: Description of feature vectors

Nr.	Description of features included	Notes	#features
1	FreeSurfer volumes for 243 regions.		243
2	FreeSurfer volumes and thickness for 70 regions.		140
3	All the above as well as all data in "label shapes" and "sulcus shapes" datasets for both left and right surface, except mean position and Laplace-Beltrami spectra.	Thickness data file contains both volume and thickness for the 70 regions, volume data for these regions was therefore added to vector twice, by mistake.	6533
4	All data in left surface "label shapes" except mean position and Laplace-Beltrami spectra.		2050
5	All data in right surface "label shapes" except mean position and Laplace-Beltrami spectra.		2050
6	All data in left surface "sulcus shapes" except mean position and Laplace-Beltrami spectra.		1025
7	All data in right surface "sulcus shapes" except mean position and Laplace-Beltrami spectra.		1025
8	FreeSurfer volumes dataset and FreeSurfer thickness from thickness dataset.	Dropping all zero valued features and normalizing data.	239
9	Same as feature vector 3, but selecting only "mean" of the statistical summaries.	Dropping all zero valued features and normalizing data.	719
10	FreeSurfer volumes	Dropping all zero valued features and normalizing data.	169

3.3.2 Getting Started: Classification of Number of APOE ε4 Alleles

Starting with feature vectors 1, 2, 4, 5, 6 and 7 (see table 3.2), RFC with default parameters was used to classify number of APOE ε4 alleles. This was a 3-way classification, as the possible labels were 0 (no alleles), 1 (one allele) and 2 (two alleles). The distribution of the class labels is shown in figure 3.2. The datasets were split into train and test sets with 0.8/0.2 distribution, and model performances were evaluated with accuracy and confusion matrix.

Same procedure were repeated for feature vector 3, though this dataset was also normalized using MinMaxScaler from scikit-learn. This dataset was then restricted using RF feature importance in the following way: one dataset with all features given importance above 0.002 and one dataset with all features given zero importance. In addition to accuracy and confusion matrix, permutation tests with 100 permutations were performed. A Dummy Classifer using stratified prediction strategy (predicting based on class distribution) was trained to use as a simple baseline.

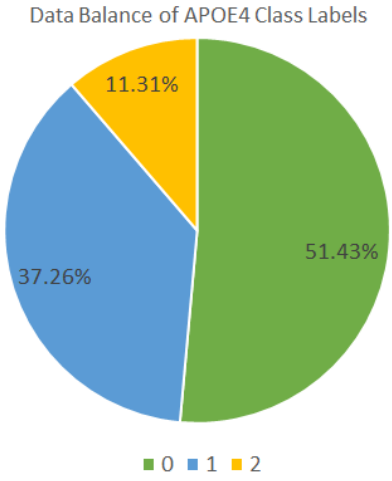


Figure 3.2: Distribution APOE ε4 class labels for the dataset

3.3.3 Feature Selection Using RF Feature Importance

To select features for the GWAS RFC experiment, recursive feature elimination and cross-validated selection (RFECV) was used with RFC model classification of AD health status of the subjects on feature vector 8, 9 and 10 (see table 3.2, using RFC importances. The dataset resulting from this feature selection will hereby be referred to as *Dataset A*. The classification of AD health status was chosen as the best way to select features as the overall goal is to find SNPs connected with developing AD. RFECV was performed on train set, and a separate test accuracy score of the model was computed. Test size of 20%. For cross validation in RFECV stratified 5-fold was used. This feature selection had two goals:

1. Finding best features
2. Finding which feature vector to use. Specifically, whether there is a benefit from using vector 9 (with 719 features) in contrast to using vector 10 (with 169 features).

As RFC feature importance seemed to heavily depend on the seed for random state, RFECV was afterwards run (on the selected feature vector from last part) for fifteen seeds, a range from 0 to 294 with step of 21, counting each time a feature was selected.

3.3.4 Using Selected Feature Dataset for RF Classification of APOE ε4 Alleles

RF classification of APOE ε4 using Dataset A. Both binary and 3-way classification were performed. The binary classification was of APOE ε4 carrier status as negative (no alleles) or positive (one or more alleles), the 3-way classification used the original labels. The data was split in train and test sets, with division 0.2/0.8 for test/train. Firstly, the parameters max depth and number of estimators for the RFC model were tuned on binary classification using stratified 5-fold cross validation of the train set. Max depths of 5, 10, 15, 20 and None, and 50, 100, 300, 500 and 1000 number of estimators were tested. The models' test performances were evaluated with accuracy score and RF feature importances displayed.

3.3.5 GWAS 1: *P*-Value Estimation of 305,479 SNPs

3.3.5.1 Criteria for Data Filtering

As stated before, the ADNI1 GWAS dataset contains genotype information for 620,901 SNPs and CNV Markers. To limit this number, the following criteria were used to filter the data:

1. SNP was to be represented by more than 1 class.
2. SNP was to have no more than 5% missing datapoints.
3. For binary data, the minority class must have at least 10% representation.
4. For multiclass data, all classes must have at least 5% representation.

This filtering resulted in dataset of 305,479 SNPs, hereby referred to as *SNP Dataset 1*.

Note: no SNP had more than 3 classes.

3.3.5.2 *P*-Value Estimation 1: Ranking by Accuracy Test Score

For every SNP, data was split into train and test set, with test set size of 20%. Test accuracy score was computed for RF classification for every SNP. Max depth of 10 was set for the RFCs, and otherwise default scikit-learn parameters (see table x). All models was then split into two sets: one with binary classifications and the other with 3-class classification. Then they were ranked from best to lowest test score and p-values were calculated by formula

$$p - value = \frac{rank}{total\ number\ of\ models}. \tag{3.2}$$

For a selected subset of these p-values, new p-values through permutation testing were computed for comparison. This subset consisted of all binary SNPs, and the permutaion testing was run with 50 permutations and stratified 3-fold cross validation.

3.3.5.3 *P*-Value Estimation 2: Ranking by Ratio between Accuracy Test Score and Majority Class Representation

Using the accuracy test scores from the latter section, the ratio between each score and their respective majority class representation was calculated, and all models were ranked from high to low using this measure (within their binary and 3-class sets). *P*-values were then calculated using equation 3.2.

The same permutation test *p*-values computed for comparison with the results from the accuracy ranking, was used for comparison with these results. For a selected subset of these *p*-values, new *p*-values through permutation testing were computed for comparison.

3.3.5.4 *P*-Value Estimation 3: Permutation Test Scores

The rough computation of permutation test *p*-values for binary SNPs in the former section, led to the idea that permutation test could be run in sequence; first do a rough estimation with few permutations to filter the SNPs, then do more accurate tests with more permutations for smaller sets. This would certainly still take time, but should be feasible. Firstly, permutation test *p*-values were therefore computed for all 305,479 SNPs, running 50 permutations with stratified 3-fold cross validation on RFC with a max depth of 10 and otherwise default parameters (see table A.1.1). From these, all SNPs with *p*-values below 0.1 were selected, hereby denoted *SNP Dataset 2*, and permutation tests with 100 permutations and stratified 5-fold cross validation was ran to compute more accurate *p*-values for this set.

Feature Dataset A were then bias corrected for age and gender using regression. The regression model for each feature was created using ols method from statsmodel with formula "feature $\sim$ age + C(gender)".

The bias corrected dataset was then used for permutation testing with 100 permutations and stratified 5-fold cross validation of all SNPs that achieved best possible permutation test *p*-value (0.00991, see equation 2.27) from the last permutation testing of SNP Dataset 2, hereby denoted *SNP Dataset 3*.

3.3.6 GWAS 2: *P*-value estimation of SNPs Located at Genes Connected to AD

A smaller dataset of SNPs from the ADNI1 GWAS dataset was extracted based on the criteria that they were located on any of the 42 genes listed at the AlzGene Database's list of top genetic risk factors of AD[4], that Ensembl[5] [124] provided chromosome, start and end positions for. This included 36 of these genes. Positional data at Ensembl uses GRCh38 as reference genome, while ADNI1 GWAS dataset uses GRCh37. The positional data was therefore converted to GRCh37. The genes and their positional data in GRCh37 coordinates can be found in table A.1.3 in Appendix 1. A SNP was termed

[4] https://web.archive.org/web/20110222233234/http://www.alzgene.org/TopResults.asp
[5] https://www.ensembl.org

"on the gene" if its position was within the start and end positions of the gene. 336 SNPs made this criteria.

Permutation tests with 7,000 permutations and stratified 5-fold cross validation, using a RFC with 100 estimators and a max depth of 10, were used to estimate p-values of these SNPs. Threshold for p-value significance set to 0.05. This was run in two parallels using data from the two different feature selections described in the following two sections. The resulting p-values were FDR corrected using the Benjamini-Hochberg procedure at level $q^* = 0.05$.

Permutation feature importance for all SNPs achieving p-values less than 0.05 from both parallels were computed using accuracy as metric, stratified 5-fold cross validation, 30 permutations per feature, with RFC model with 100 estimators and max depth of 10.

3.3.6.1 First Permutation Feature Importance Feature Selection

Data of feature vector 10 was bias corrected for age and gender following same procedure as described in section 3.3.5.4, and then normalized. Using AD status as labels, RFECV with stratified 3-fold cross validation and a minimum of 20 features to be selected, was then performed to select features using permutation feature importance. The permutation feature importance model was set to run 10 iterations (for each feature), using accuracy as metric, and stratified 3-fold cross validation on a RFC model (default parameters). The dataset from this feature selection will hereby be denoted *Dataset X*.

3.3.6.2 Second Permutation Feature Importance Feature Selection

To avoid the weakness of correlated features getting weaker scores in permutation feature importance, pairwise Pearson correlation coefficients between features was computed for the age and gender bias corrected vector 10 dataset. Each set of correlated features was then represented with only one feature such that the resulting set of features was of not correlated features (one feature could be present in many sets). Two features was considered correlated if the absolute value of their correlation coefficient was greater than 0.6. The set of representative SNPs and correlated partners are given in table A.1.4 in Appendix 1.

As all correlated features would be appended to the selected features, more control over how many features were selected than RFECV provides was needed. Recursive feature elimination (RFE) set to select 15 features was therefore chosen. The RFE was set to remove one feature at each iteration (step of 1) and use permutation feature importance for the selection. The permutation feature importance model was set to run stratified 5-fold cross validation with 20 permutations per feature, and used accuracy as scoring metrics. RF classification of AD health status with default parameters was used for the selection.

To the set of the 15 selected features, their correlated features were added. This set will hereby be denoted *Dataset Y*.

3.3.7 APOE ε4 Carrier Status Classification

Random Forest Classification of binary APOE ε4 carrier status (0=no alleles, 1=one or more alleles), using feature Dataset Y. Data was split into train and test sets with test set size of 20%. Number of estimators and max depth of RFC model was tuned by running stratified 7-fold cross validation with accuracy as metric on the train set. Testing 100, 300, 500 and 1000 estimators and max depths of 5, 10, 15, 20 and none. Using the best scoring parameters for the RFC model, the model was trained, and accuracy test score and confusion matrix was computed. Permutation feature importance of the model was computed on the test set (using the "prefit" setting), using 50 permutations per feature and accuracy as metric. Classification performance was also evaluated with permutation testing, running 1000 permutations, accuracy as metric, with stratified 7-fold cross validation on the train set.

3.4 Phase 2: Deep Learning in Imaging Genetics

As it is a relatively fast model to train and there exists ready implementations for 3D image classifications, the CNN architecture DenseNet was chosen for all DL classification tasks. More specifically, DenseNet-121, as Ruiz et al. found in 2020 that it had both the best performance and most efficient training time of the DenseNet depths, in classification of AD from 3D brain MR images [98]. The following hyper-parameter settings were used while training:

- Adam stochastic optimization algorithm with learning rate set to 0.0001, from PyTorch.
- Cross-Entropy loss function, from PyTorch.
- Batch size of 4.
- Dropout probability of 0.5.
- Growth rate of 32.
- Training was run for 50 epochs.

3.4.1 APOE ε4 Carrier Status Classification with DenseNet

One of the greatest challenges when it comes to training CNN models for this classification task is the limited amount of images to train on. The effects of pretraining have therefore been studied in this part by running 4 parallels for classification of binary APOE ε4 carrier status:

1. Training DenseNet model in classification of APOE ε4 directly, with no pretraining.

2. First training a DenseNet model in 5-way classification of AD, then carrying over the weights from this to the training in classification of APOE ε4.

3. Firstly, training a DenseNet in classification of gender. Secondly, using weights from this pretraining for model training in 5-way classification of AD. Finally, using the last weights in training model in classification of APOE ε4.

4. First training a DenseNet model in 3-way classification of AD, then using these weights for classification of APOE ε4.

All dataset splits into train, validation and test sets, has followed the criteria that all images for the same test subject is in the same set and that the data label balance is kept in all sets. The full dataset consisted of 7,372 MR images. Firstly, the dataset was split into train/-validation/test with sizes 5,871/749/741 (approx. 80%/10%/10%) for APOE ε4 classification. Some images had to be dropped for this classification as there was no information on APOE ε4 for the subject. This test set was held out for all classifications, so that it was completely "unseen" for the final model classification in all four parallels. The remaining dataset splits are therefore done on the dataset of $7,372 - 741 = 6,631$ images.

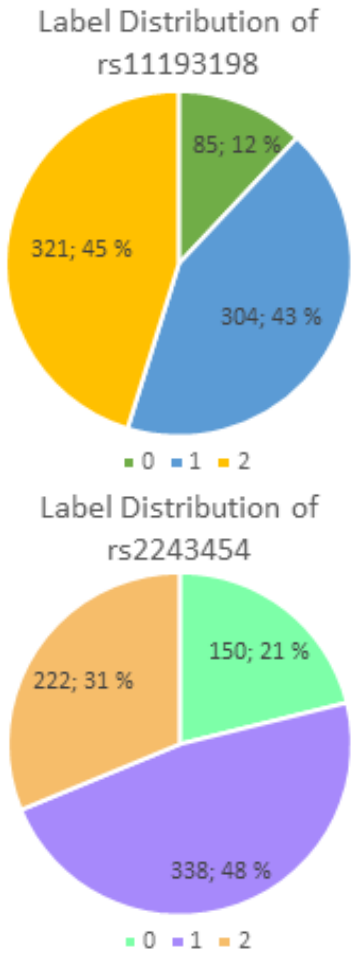


Figure 3.3: Distribution of rs11193198 and rs2243454 class labels per subject.

The 5-way classification of AD health status has labels "AD", "CN", "EMCI", "LMCI" and "SMC" (see section 3.1.1). Train/validation/test set sizes after splitting of 5,552/539/540. For the 3-way classification of AD, the label "SMC" was dropped, and "EMCI" and "LMCI" was merged to "MCI". Train/validation/test set sizes after splitting of 5,401/517/518. The train/validation/test sizes for classification of gender were 5,567/533/531.

All models were evaluated with classification accuracy and confusion matrix. All the binary classifications (gender and APOE ε4) were also evaluated with precision, recall and F1 score.

3.4.2 3D DenseNet Classification of rs11193198 and rs2243454

The next step of this phase was to try deep learning classification of a few SNPs. For this task rs11193198 and rs2243454 on the SORCS1 gene were selected. Why these two SNPs were selected will be explained later. The best performing pretraining (or no

pretraining) pipeline from last section was followed for these classifications.

As there are genotype data for fewer subjects (710), the complete dataset for these tasks consisted of 3,561 images. The data was split into train, validation and test sets for both SNPs, with respective sizes of 2,842/358/361 for rs11193198 and 2,837/362/362 for rs2243454. Criteria for the splitting were the same as in last section, i.e. that label balance was kept in all sets and that all images for a subject was in the same set. The distribution of the classes with regard to subjects, of the two SNPs are shown figure 3.3.

3.4.3 Occlusion Sensitivity

Occlusion sensitivity maps were computed for some test images for one AD model, one APOE $\varepsilon 4$ model, and the gender, rs11193198 and rs2243454 models. First, the effect of stride and mask size were tested for the selected AD model for image ID I223061 of subject 067_S_2304, which has label "EMCI". Stride must be a factor of image size, in our case of $128 \times 128 \times 128$, and stride and mask size should be element-wise either odd or even. As the only odd factor of 128 is 1, odd mask/stride sizes were avoided, as strides of 1 requires $128 \cdot 128 \cdot 128 = 2,097,152$ tests, compared with $32 \cdot 32 \cdot 32 = 32,768$ for strides of 4 and $16 \cdot 16 \cdot 16 = 4,096$ for strides of 8. The following mask size/stride size couplings were tested:

- $(8 \times 8 \times 8)/(8 \times 8 \times 8)$
- $(16 \times 16 \times 16)/(4 \times 4 \times 4)$
- $(16 \times 16 \times 16)/(8 \times 8 \times 8)$
- $(16 \times 16 \times 16)/(16 \times 16 \times 16)$
- $(24 \times 24 \times 24)/(4 \times 4 \times 4)$
- $(24 \times 24 \times 24)/(16 \times 16 \times 16)$
- $(32 \times 32 \times 32)/(32 \times 32 \times 32)$

The most informative mask/stride size coupling was then used to compute occlusion maps for the following images:

- For the AD model:
  - I112212 of subject 022_S_1097, label "LMCI"
  - I118840 of subject 023_S_0031, label "CN"
  - I280549 of subject 019_S_4252, label "AD"
  - I402028 of subject 057_S_5295, label "SMC"
- For the APOE model:
  - I270045 of subject 068_S_4340, label 0
  - I374493 of subject 002_S_5018, label 1

- For the rs11193198 model:
  - I33644 of subject 021_S_0424, label 0
  - I79766 of subject 037_S_0467, label 1
  - I118838 of subject 018_S_0087, label 2
- For the rs2243454 model:
  - I79373 of subject 073_S_0518, label 0
  - I65343 of subject 027_S_0850, label 1
  - I39108 of subject 130_S_0232, label 2
- For the Gender model:
  - I105474 of subject 116_S_0361, label male
  - I143615 of subject 021_S_0159, label female

4 Results and Analysis

4.1 Phase 1: Random Forest Classifier in Imaging Genetics

4.1.1 Getting Started: Classification of Number of APOE ε4 Alleles

The point of this section was merely to get started and get an understanding of the data and the tasks at hand. The accuracy test scores in table 4.1 and 4.2 are approximately the same as the DummyClassifier performance. However, from the confusion matrices in figure 4.1 and 4.3, it is clear that they are not simply predicting from the class distribution, as none predicts label 2. The permutation tests on all, important and insignificant features, (table 4.2 for *p*-values and figure 4.2 for scores) strongly suggests that the RFCs learn a structure in the data, and that feature selection leads to significant improvements for these classification tasks.

The feature importance display (figure 4.2) led to the realization that the datasets contained many zero-valued features, and consequently these were removed for future experiments, resulting in the datasets feature vector 8, 9 and 10 (see table 3.2).

Table 4.1: Accuracy test scores for the RFC models. Feature vector number gives which dataset was used for training the model (see table 3.2).

Vector	**1**	**2**	**4**	**5**	**6**	**7**
Accuracy	46.03%	49.21%	46.83%	50.79%	52.38%	48.41%

Table 4.2: Accuracy test, cross validation and p-value scores for each RFC model trained with different sets of features from feature vector 3, in addition to the accuracy test score of the DummyClassifier in classification of APOE ε4.

Features	**All**	**Important**	**Insignificant**	**Dummy**
Test accuracy score	43.65%	52.38%	46.83%	47.62%
Cross validation score	50.79%	51.74%	48.56%	
***p*-value**	0.1782	0.01980	0.7624	

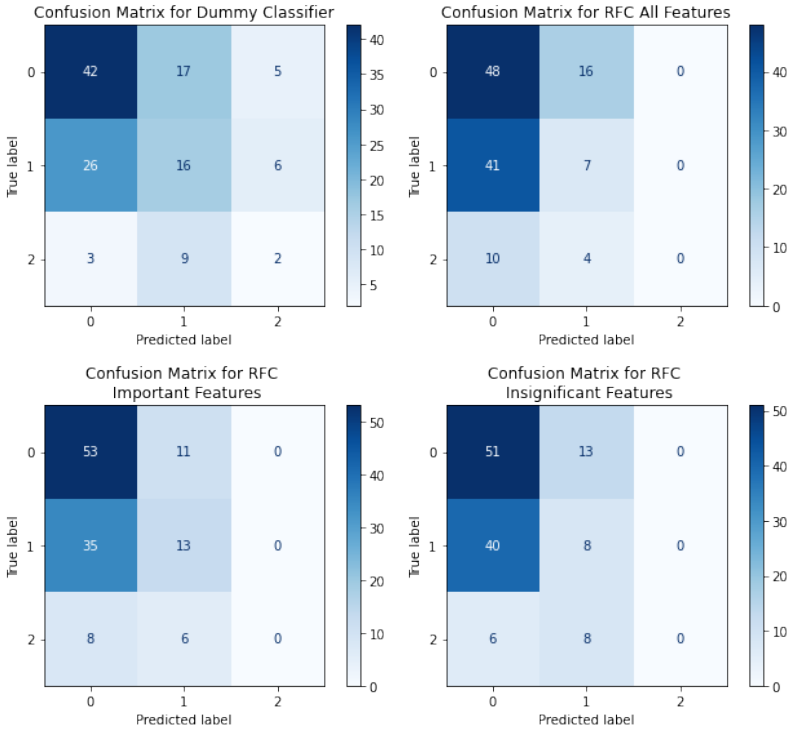


Figure 4.1: Confusion matrices for the dummy classifier and each RFC model trained with different sets of features from feature vector 3, classification of APOE ε4.

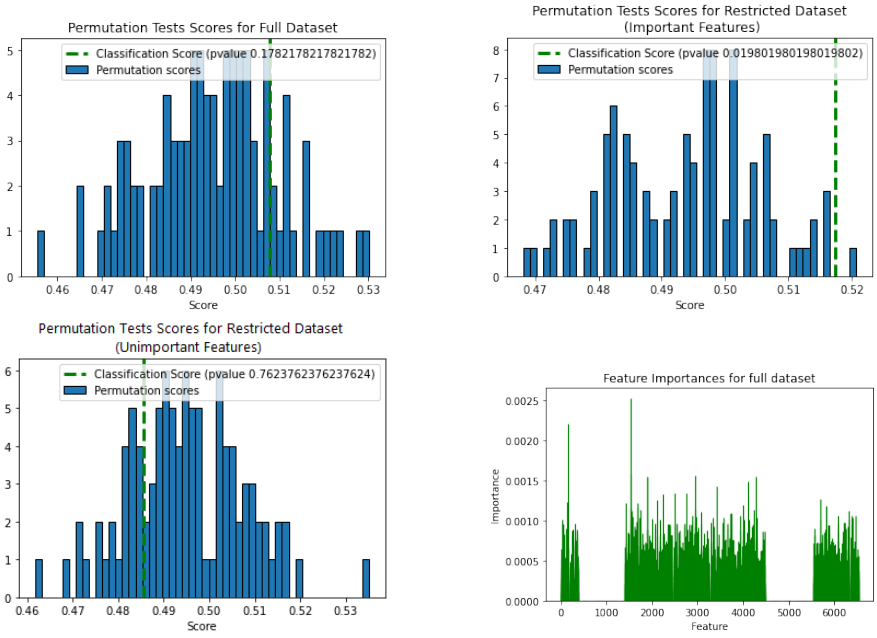


Figure 4.2: Top and bottom left: Permutation test scores for each RFC model trained with different sets of features from feature vector 3, in classification of APOE ε4. Bottom right: RF feature importances for the RFC model with the full dataset.

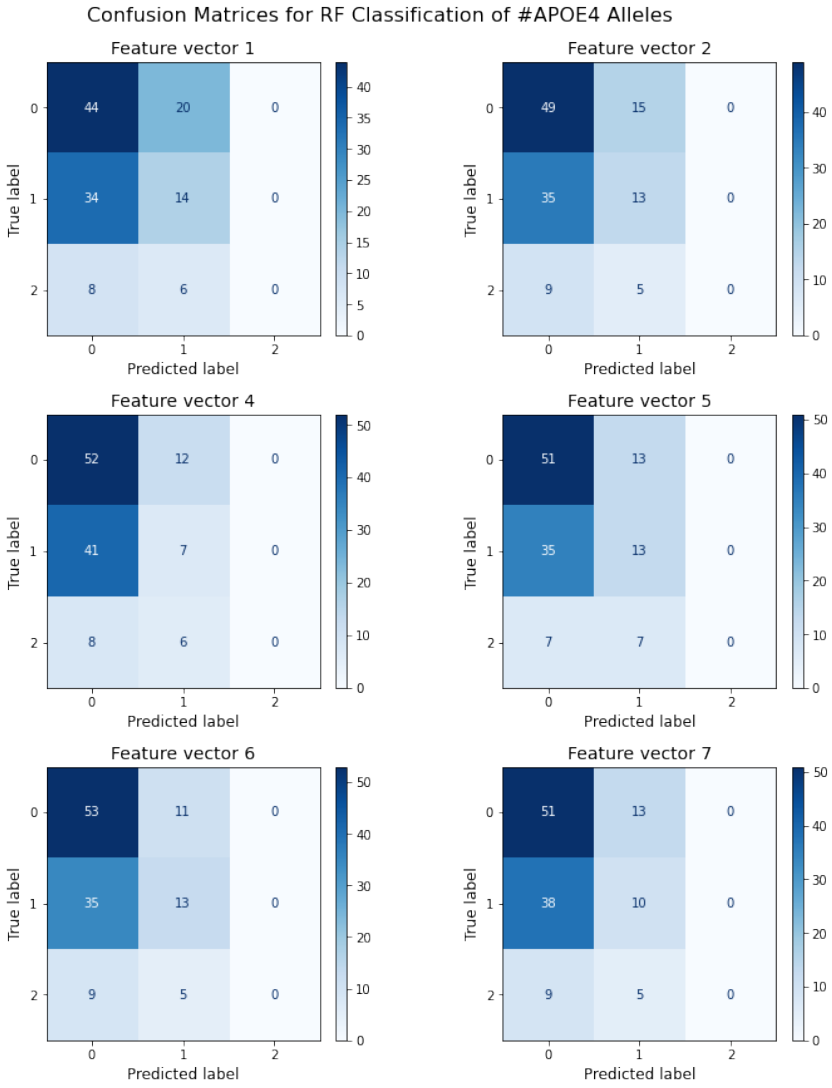


Figure 4.3: Confusion matrices for each RFC model, classification of APOE ε4.

4.1.2 Feature Selection Using RF Feature Importance

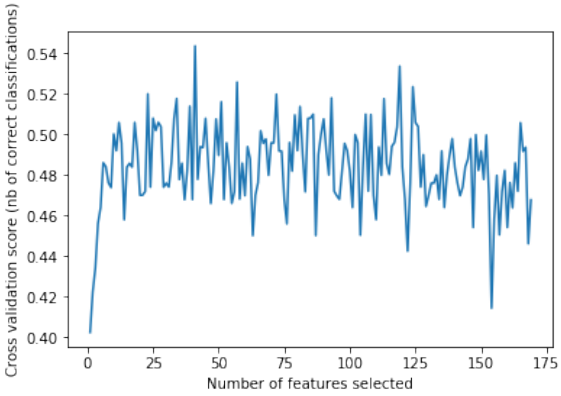
The most important aspects of this feature selection was:

1. to select relevant features for the ML task.
2. to avoid having too many features for training efficiency.

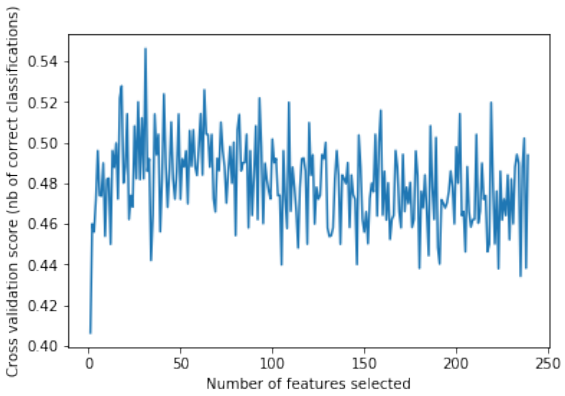
An important question was therefore: *is there a point in using the datasets with all possible features (feature vector 9), or is using just one measure (feature vector 10) enough?* From table 4.3, it can be deduced that the performance of the models are approximately the same. The cross validation scores from the RFECVs are shown in figure 4.4. The general trend of all of these RFECVs are the same, and lies between approx. 0.46 and 0.51. The RF feature importances for the selected features of feature vector 8 and 10 are shown in figure 4.2. Only two of the selected features were not FreeSurfer volume features. The three highest scoring features of feature vector 10 are from hippocampus and amygdala areas, two brain regions known to be early affected by AD. The chosen dataset (called Dataset A) from this feature selection was therefore the 41 RFECV selected features from feature vector 10.

Table 4.3: The number of features selected from RFECV and accuracy test scores for RF classification of AD health status, of models trained with the selected features and all features for the given feature vector numbers (see table 3.2).

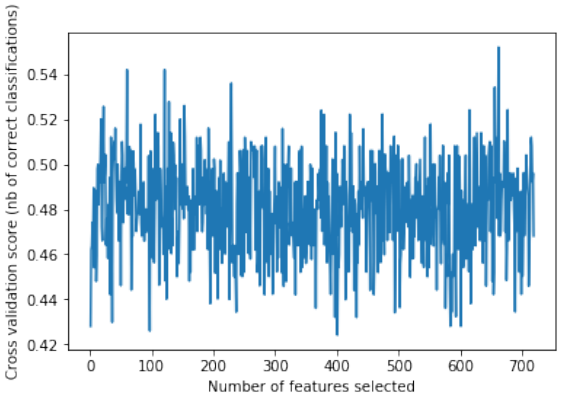
Vector	8	9	10
Number of features selected	31	662	41
Accuracy for all features	58.73%	59.52%	58.73%
Accuracy for selected features	59.52%	61.90%	56.38%



(a) Feature vector 10



(b) Feature vector 8



(c) Feature vector 9

Figure 4.4: RFECV results for RFCs trained on the given datasets.

As stated in section 3.3.3, the RFECV feature selection seemed to heavily depend on the seeding. RFECV of feature vector 10 was therefore tested with 15 seed, counting each time a feature was selected. Though all, except one, was selected at least once, only 13 was selected 12 times or more (hippocampus volumes amongst them), and 44 (top 13 included) more than half the times. Thus, 125 features were selected less than half the times, which strongly suggests that not all the features are selected by chance, but some are. This is however, from looking at the RFECV results seen in figure 4.4a, expected, as the maxima and minima of the RFECV results seem arbitrary given the general trend. For one seed, 41 features could be selected, while for another, it could

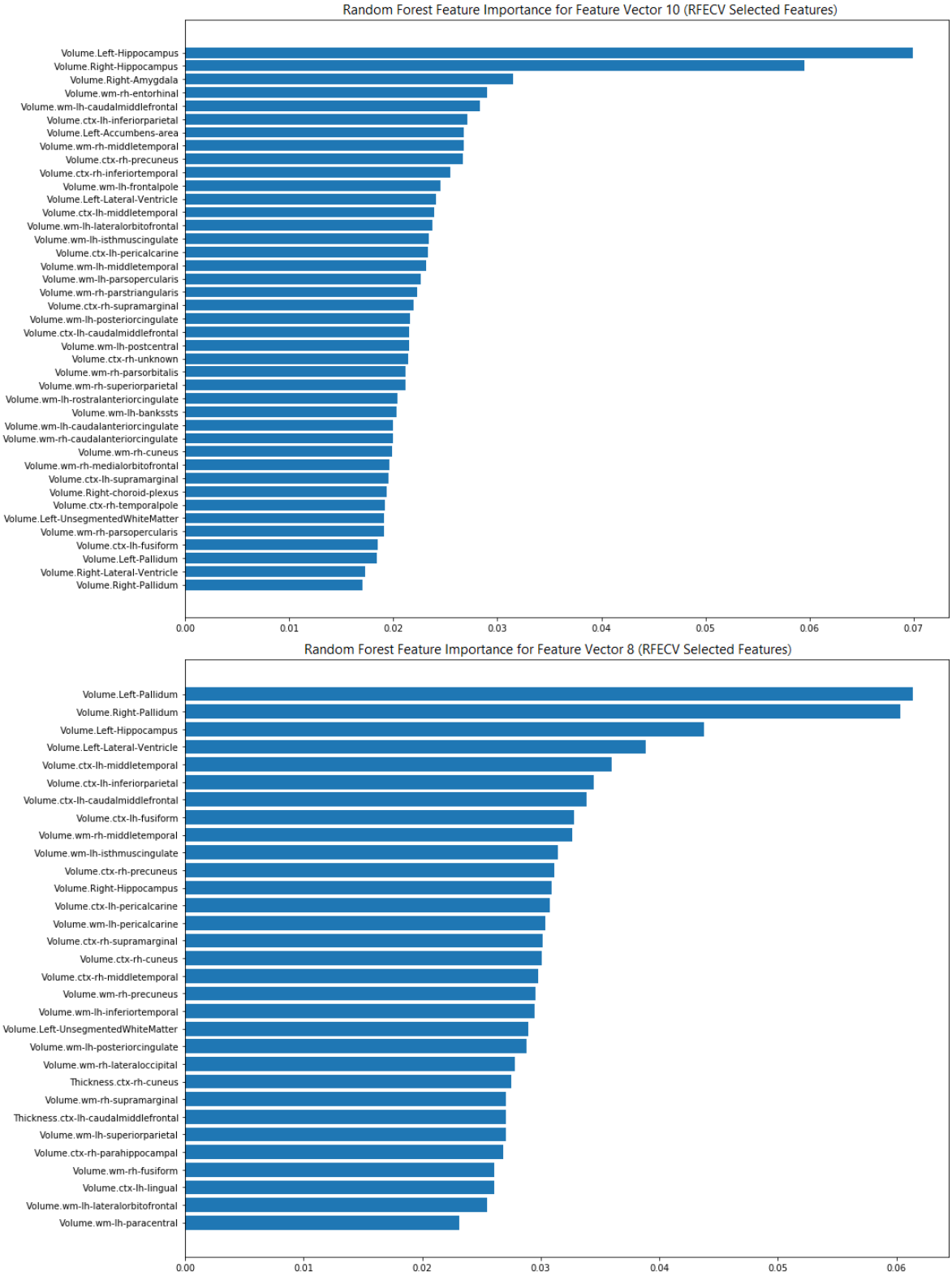


Figure 4.5: Random Forest Feature Importance for the selected features of feature vector 8 and 10.

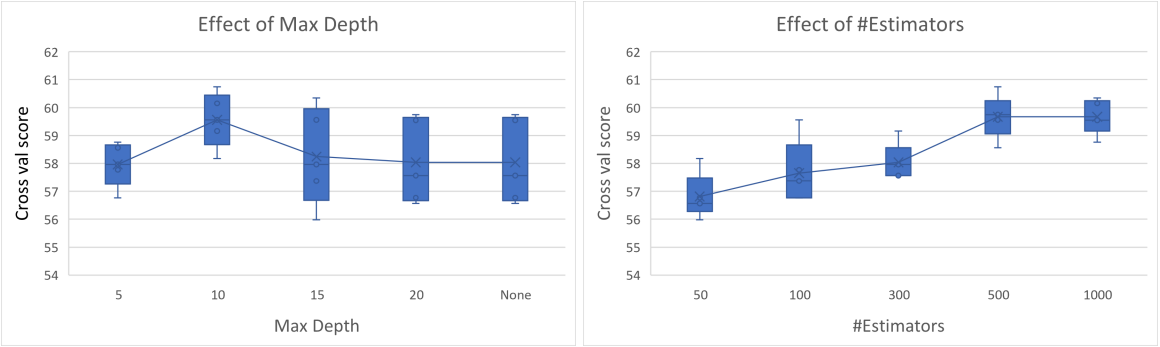
easily be 154. This can be seen as an indication of the unreliability of RF feature importance, discussed in section 2.3.1.1.

4.1.3 Using Selected Feature Dataset for RF Classification of APOE ε4 Alleles

Figure 4.6 displays all cross validation results from the hyper-parameter tuning. The selected parameters were 500 estimators and max depth of 10. Both the number of estimators and the max depth clearly have an effect on the results, though there is no difference in the results between no limit to the depth and a limit of 20. This might indicate that with no limit it does not pass 20. As a max depth of 10 improves performance for this model (figure 4.6a), and limits the training time for RFC models, it has been used as a standard for the rest of the experiments.

The accuracy score on test data was 65.87% for the binary classification and 57.14% for the 3-way classification. That binary classification have better performance is expected. From the confusion matrices in section 4.1.1 (figures 4.3 and 4.1), it is clear that the classifiers struggle to separate classes 1 and 2.

That the RF feature importances for hippocampus volumes are highest for both binary and 3-way classification is promising, further on that amygdala volumes are second highest for binary, and in the top 6 for 3-way (see figure 4.7).



(a) Max Depth *(b) Number of Estimators*

Figure 4.6: Results from hyper-parameter tuning of RFC model for binary classification of APOE ε4.

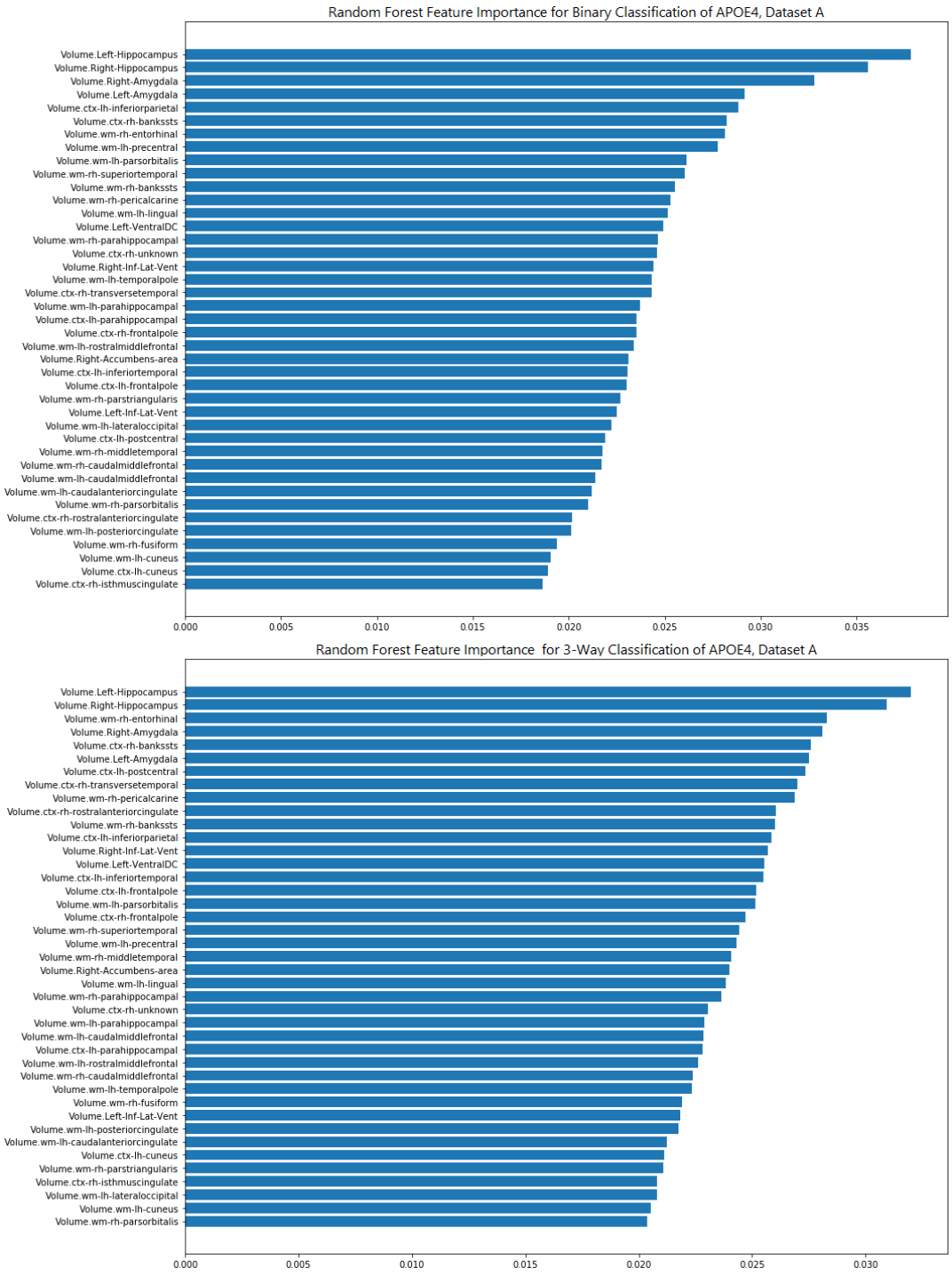
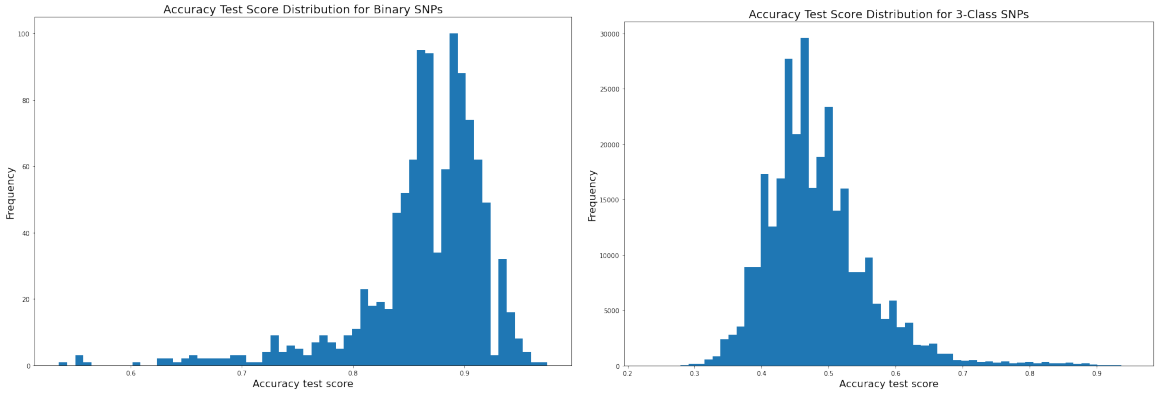


Figure 4.7: Random Forest Feature Importance for Classification of APOE ε4

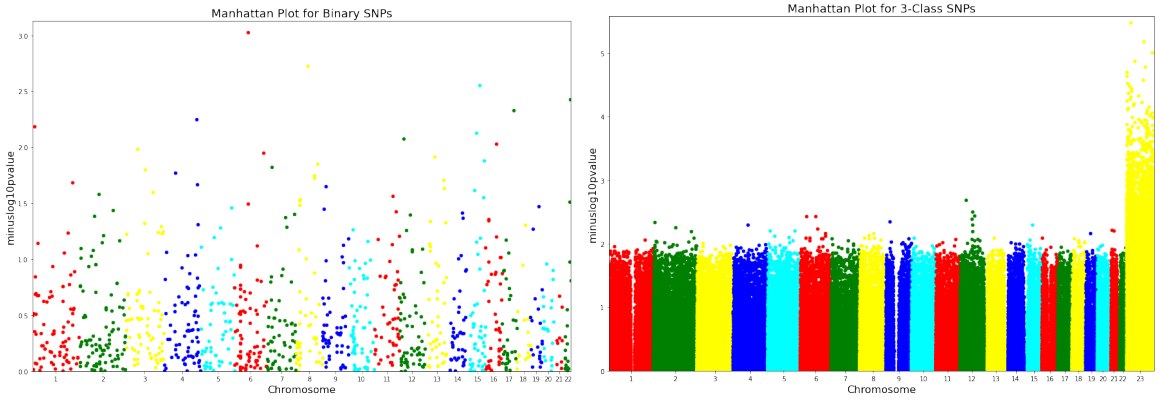
4.1.4 GWAS 1: *P*-Value Estimation of 305,479 SNPs

4.1.4.1 *P*-Value Estimation 1 and 2: Ranking by Accuracy Test Score and by Ratio between Accuracy Test Score and Majority Class Representation

The accuracy test scores distributions shown in figures 4.8a and 4.8b, look promising for this way of computing p-values. However, the comparison of rank with permutation tests p-values in figure 4.9a showed no correlation. Considering that this was due



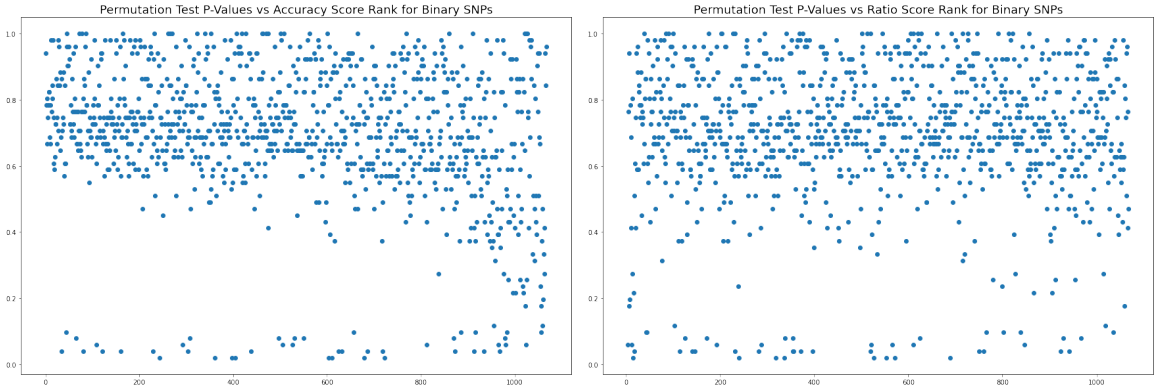
(a) Binary *(b) 3-Class*



(c) Binary *(d) 3-Class*

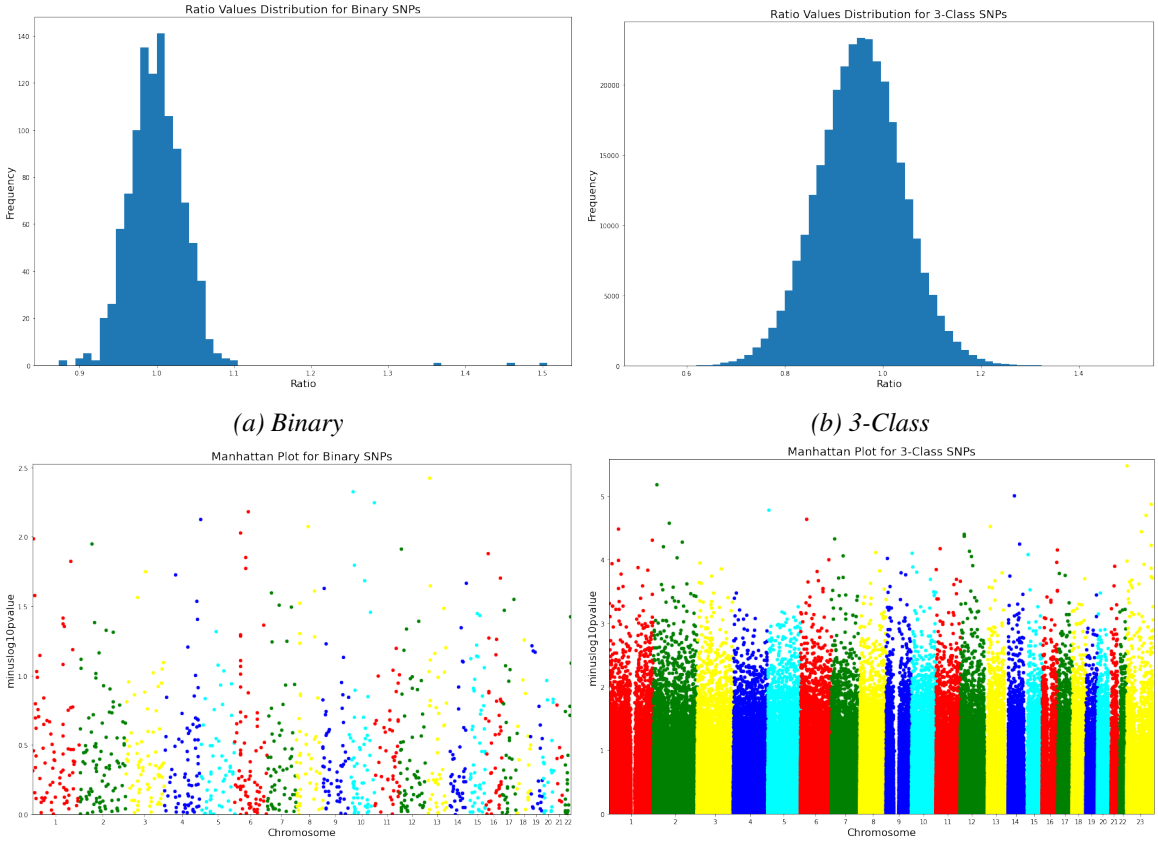
Figure 4.8: Accuracy test score distributions and Manhattan plots for binary and 3-class SNPs, p-values estimated using rank based on accuracy.

to differences in class label balance for the SNPs, the idea to use the ratio between the accuracy score and the maximum class representation was tested. These distributions, showed in figure 4.10a and 4.10b, were even more promising. Unfortunately, this way of ranking the SNPs neither showed any correlation with the permutation test p-values, as seen in figure 4.9b.



(a) Accuracy test score ranking *(b) Ratio score ranking*

Figure 4.9: Permutation test score p-values for binary SNPs, plotted against their respective accuracy test score or ratio ranking.



(a) Binary *(b) 3-Class*

(c) Binary *(d) 3-Class*

Figure 4.10: Ratio distributions and Manhattan plots for binary and 3-class SNPs, p-values estimated using rank based on ratio between accuracy test scores and majority class representation.

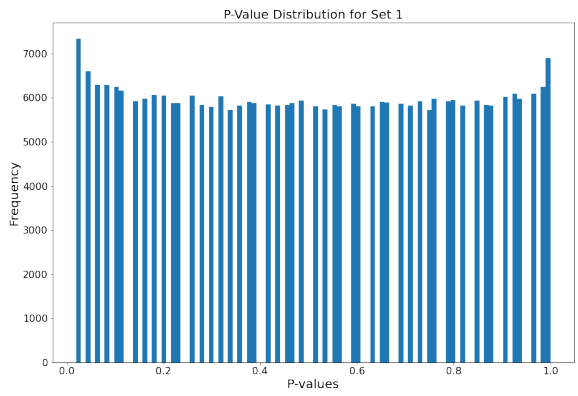
While not much information can be gathered from the Manhattan plot of ratio *p*-values shown in figure 4.10d, it is evident from the Manhatton plot of accuracy score *p*-values in figure 4.8d that all of the highest scoring SNPs belong to chromosome X (23 in the graph). As this is not the case for the ratio Manhattan plot, it is highly likely that the SNPs on chromosome X have a great majority class that the classifiers are learning to predict.

4.1.4.2 *P*-Value Estimation 3: Permutation Test Scores

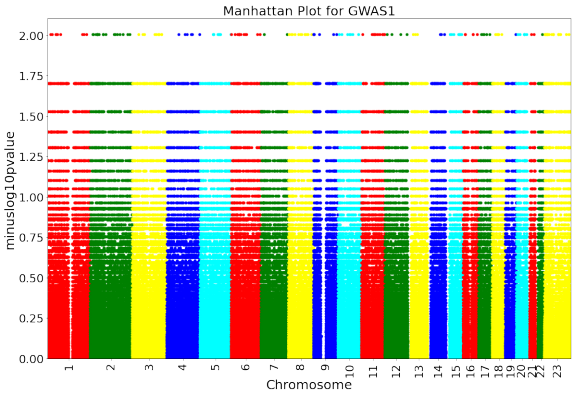
32,770 SNPs achieved a smaller *p*-value than 0.1 in the permutation testing of SNP Dataset 1, and was consequently appended to SNP Dataset 2. Of the permutation testing of SNP Dataset 2, 2,289 SNPs achieved the best possible *p*-value of 0.009901 and was consequently appended to Dataset 3.

As expected, computing more accurate *p*-values by running more permutations displays that some of the results from the first, rough permutation tests are false discoveries. This effect is displayed in figure 4.12a, where most of the SNPs still get *p*-values below 0.1, but the distribution is elbow shaped, with some SNPs even scoring worst possible *p*-value of 1.

The over representation of chromosome X SNPs in the distribution of best scoring *p*-

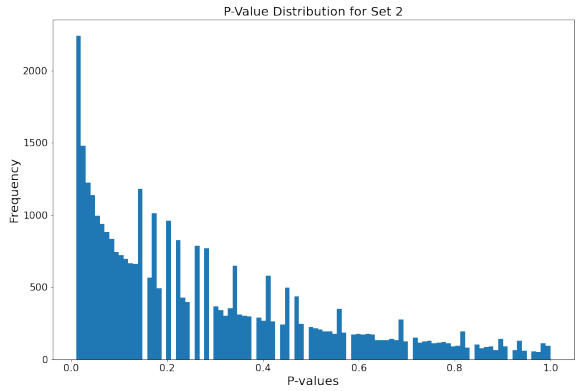


(a) P-value distribution of set 1 (first permutation tests of all SNPs)

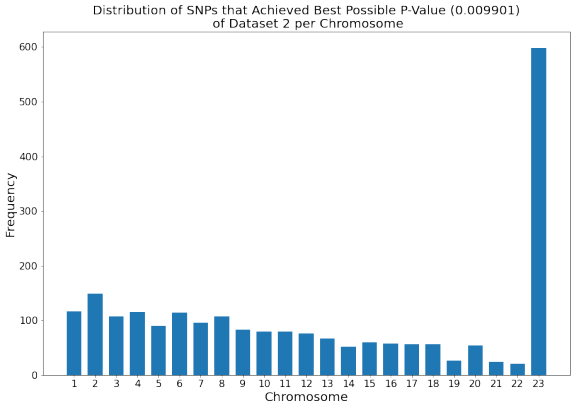


(b) Manhattan plot for final results of all SNPs (set 1, 2 and 3 combined)

Figure 4.11: The distribution and Manhattan plot of permutation test p-values for all 305,479 SNPs.



(a) Distribution of p-values.



(b) Distribution of best p-values per chromosome.

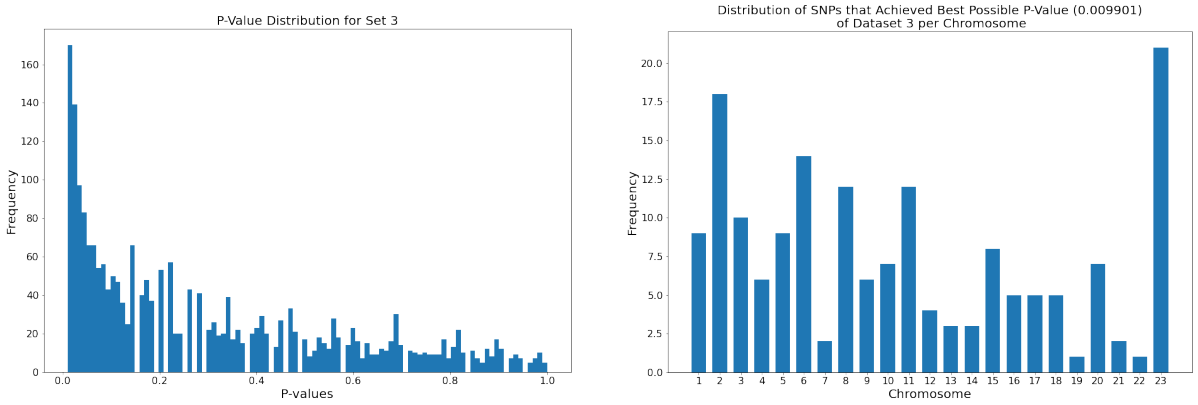
Figure 4.12: The distribution of permutation test p-values for SNP Dataset 2 and the distribution per chromosome of all SNPs that achieved the best possible p-value score of 0.009901.

values of set 2, shown in figure 4.12b, led to the realisation that bias correcting the features would possibly be a good idea. The features were therefore corrected for age and gender and the third permutation testing were performed.

After the third permutation testing with bias corrected features, 170 of the 2,289 SNPs got a *p*-value of 0.009901. Clearly, the bias correction had an effect, though the *p*-value distribution of this permutation testing shown in figure 4.13a is also elbow shaped, with the highest density <0.05. The distribution of top scoring SNPs per chromosome is now more spread. Though chromosome X is still the most represented, this is expected as it has the highest representation over all in SNP set 3.

Not much information can be gained from the Manhattan plot of all final *p*-values in 4.11b. Naturally, it is discrete, as the possible *p*-values from permutation testing are discrete. However, of the top scoring SNPs, there is nothing indicating that they belong to specific areas of the chromosomes.

The computational time of these permutation tests (approximately two weeks running



(a) Distribution of p-values. *(b) Distribution of best p-values per chromosome.*

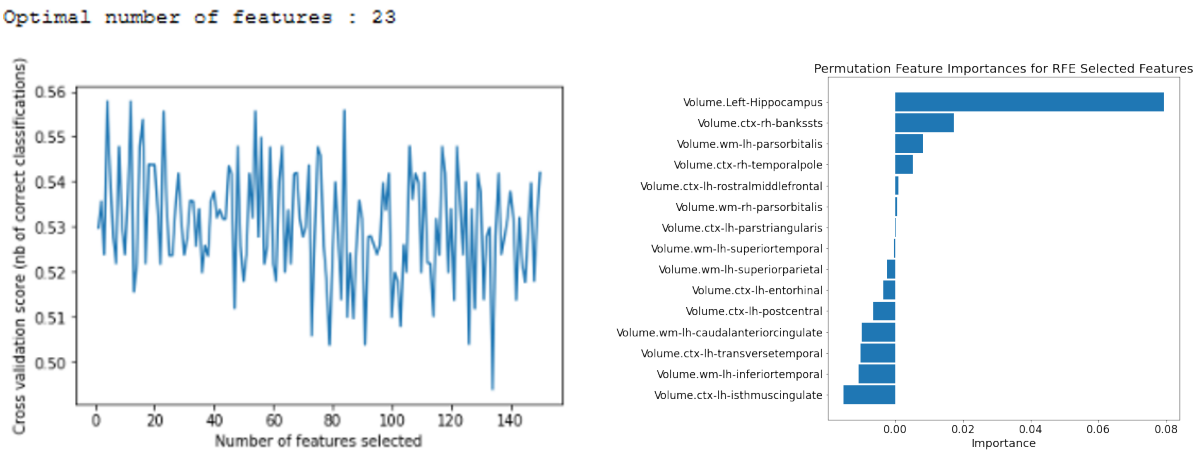
Figure 4.13: The distribution of permutation test p-values for SNP Dataset 3 and the distribution per chromosome of all SNPs that achieved the best possible p-value score of 0.009901.

parallel on multiple CPUs), led to the decision to select the smaller set of SNPs belonging to genes already found to be associated with AD, so as to better estimate whether the RF classifications successfully finds data structures of interest.

4.1.5 GWAS 2: *P*-value estimation of SNPs Located at Genes Connected to AD

4.1.5.1 The Feature Selections

Taking the instability of RF feature importance into consideration, new feature selections were performed with permutation feature importance. The RFECV results from the first selection is displayed in figure 4.14a. The resulting Dataset X contained 23 features selected by the RFECV.

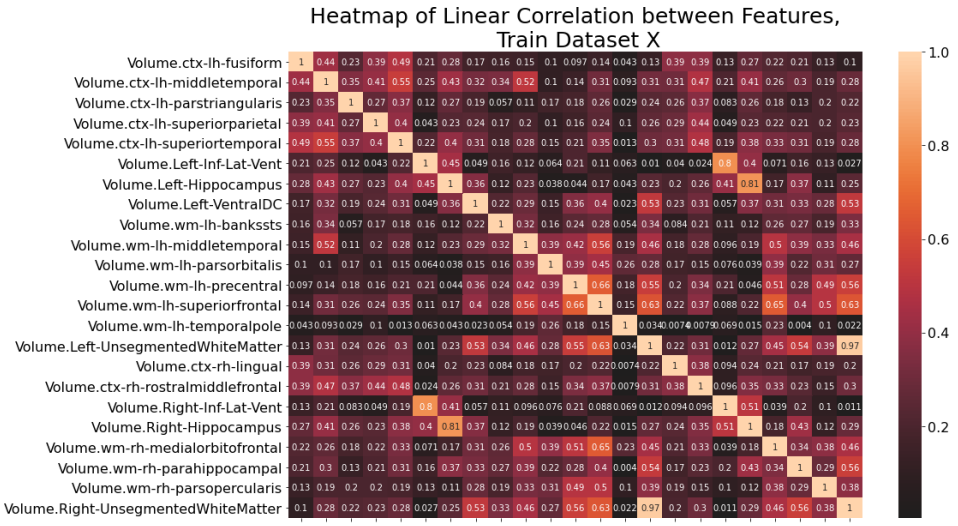


(a) Results from the RFECV of the first feature selection. *(b) Permutation feature importances for the RFE selected features.*

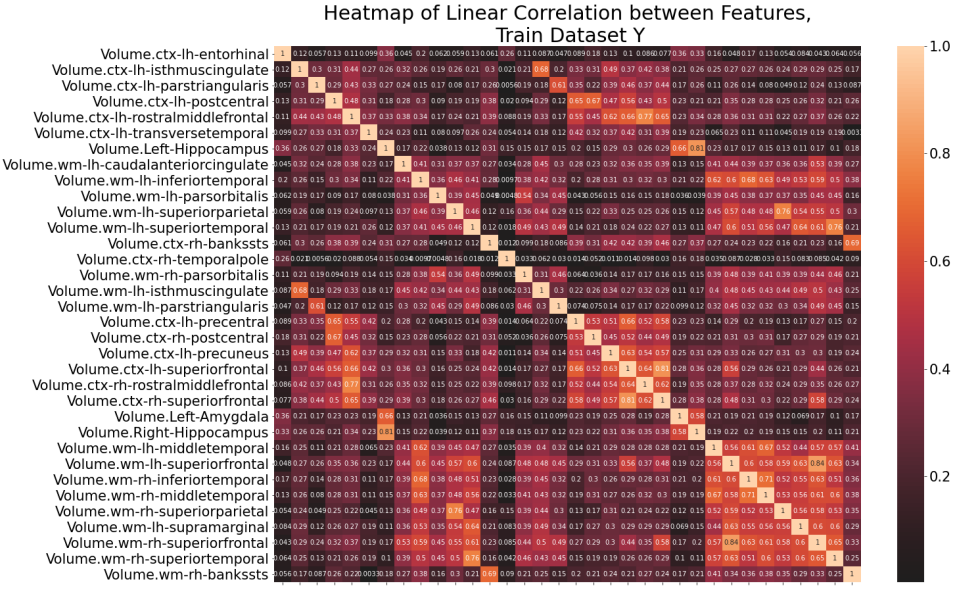
Figure 4.14: The results from the RFECV of the first feature selection (making Dataset X), and the permutation feature importances for the 15 RFE selected features from the second feature selection (making Dataset Y).

For the last feature selection, the creation of Dataset Y, the weakness of permutation feature importance with regard to feature correlation was taken into account. The set of representative features (see table A.1.4, Appendix 1) contained 92 features. With the appended correlated features to the selected 15, Datset Y ended up with 34 features. As seen in figure 4.14b, many of the selected 15 features received negative importance. It can be argued that they could have been omitted from the dataset, or rather, that the RFE selection could be continued until all features only received positive scores.

The feature correlation of Datset X and Dataset Y can be seen in figure 4.15. Naturally, Dataset Y contains a lot more correlated features than Dataset X. The sets also ended up drastically different, with only seven features in common.



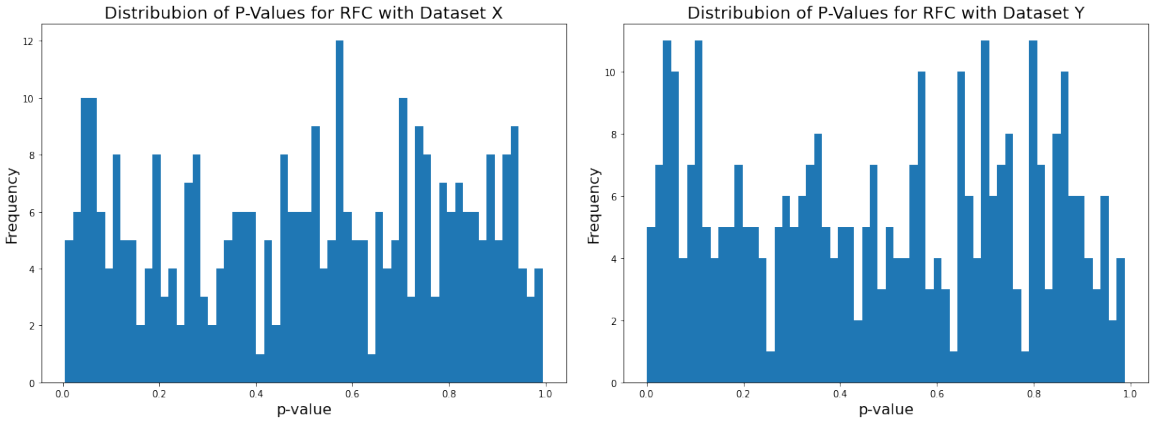
(a) Dataset X



(b) Dataset Y

Figure 4.15: Pairwise Pearson correlation coefficients for all features of Dataset X and Dataset Y. The labels on the x-axis are the same as on the y-axis.

4.1.5.2 The Permutation Tests



(a) Dataset X *(b) Dataset Y*

Figure 4.16: P-Value distributions from the permutation tests of Dataset X and Dataset Y.

The p-value distributions of the permutation tests of Dataset X and Dataset Y are shown in figure 4.16a and 4.16b, respectively. Of Dataset X, 20 SNPs beat the 0.05 threshold, displayed in table 4.4. Of Dataset Y, 22 SNPs beat the 0.05 threshold. After FDR correction, none of the *p*-values remained significant for either of the permutation test parallels.

Table 4.4: Data on all SNPs achieving p-value less than 0.05 in permutation tests of Dataset X, ordered from best to worst. Columns 1. p-value and 1. cross-val score indicate the p-value and cross validation accuracy score from this permutation tests of Dataset X, respectively. Columns 2. p-value and 2. cross-val score indicate the p-value and cross validation accuracy score from the latter permutation tests of Dataset Y, for comparison. The differences in p-value and cross validation scores are given as scores from Dataset Y subtracted from scores from Dataset X.

Nr.	SNP	Gene name	1. p-value	1. cross-val score	2. p-value	2. cross-val score	diff p-value (1. - 2.)	diff cross-val score(%) (1.-2.)
1	rs17126035	SORL1	0.004995	0.604055	0.050421	0.590439	-0.045426	1.361576
2	rs10519085	ADAM10	0.008991	0.491350	0.349807	0.465411	-0.340816	2.593982
3	rs11193198	SORCS1	0.010989	0.491394	0.006999	0.491394	0.003990	0.000000
4	rs4844851	CR1	0.017982	0.495844	0.079274	0.478409	-0.061292	1.743578
5	rs10519074	ADAM10	0.017982	0.527454	0.160120	0.513556	-0.142138	1.389805
6	rs1544267	GAB2	0.021978	0.520557	0.099272	0.500000	-0.077294	2.055703
7	rs1120599	CR1	0.022977	0.515459	0.089559	0.506838	-0.066582	0.862069
8	rs12420711	GAB2	0.023976	0.474768	0.447222	0.437250	-0.423246	3.751765
9	rs12420296	GAB2	0.026973	0.442124	0.546065	0.412691	-0.519092	2.943339
10	rs1336978	SORCS1	0.028971	0.448479	0.326096	0.422763	-0.297125	2.571596
11	rs2243454	SORCS1	0.036963	0.460593	0.025853	0.465575	0.011110	-0.498233
12	rs6584766	SORCS1	0.038961	0.482682	0.196543	0.470877	-0.157582	1.180520
13	rs950809	SORCS1	0.040959	0.525732	0.120126	0.522269	-0.079167	0.346327
14	rs717751	SORCS1	0.041958	0.541292	0.350950	0.520573	-0.308992	2.071888
15	rs1351545	ADAM10	0.042957	0.574050	0.918726	0.531050	-0.875769	4.300004
16	rs1790557	GAB2	0.047952	0.508373	0.195543	0.487871	-0.107740	1.358680
17	rs3905934	TF	0.047952	0.638635	0.614769	0.623147	-0.566817	1.548878
18	rs6908326	NEDD9	0.047952	0.572195	0.557777	0.548130	-0.509825	2.406528
19	rs755577	CALHM1	0.047952	0.522134	0.155692	0.508547	-0.147591	2.050161
20	rs1986011	PCDH11X	0.048951	0.436624	0.296243	0.412426	-0.247292	2.419893

Table 4.5: Data on all SNPs achieving p-value less than 0.05 in permutation tests of Dataset Y, ordered from best to worst. Columns 2. p-value and 2. cross-val score indicate the p-value and cross validation accuracy score from this permutation tests of Dataset Y, respectively. Columns 1. p-value and 1. cross-val score indicate the p-value and cross validation accuracy score from the former permutation tests of Dataset X, for comparison. The differences in p-value and cross validation scores are given as scores from Dataset Y subtracted from scores from Dataset X.

Nr.	SNP	Gene name	1. p-value	1. cross-val score	2. p-value	2. cross-val score	diff p-value (1. - 2.)	diff cross-val score(%) (1.-2.)
1	rs744373	BIN1	0.058941	0.495512	0.001286	0.527283	0.057656	-3.177019
2	rs7101566	SORL1	0.906094	0.871143	0.003857	0.878010	0.902237	-0.686682
3	rs11193198	SORCS1	0.010989	0.491394	0.006999	0.491394	0.003990	0.000000
4	rs292092	PICALM	0.701299	0.429618	0.013569	0.486414	0.687729	-5.679545
5	rs9380116	NEDD9	0.732268	0.463341	0.014427	0.524410	0.717841	-6.106868
6	rs6676300	MTHFR	0.186813	0.440774	0.019997	0.468107	0.166816	-2.733336
7	rs642378	GAB2	0.139860	0.475965	0.021568	0.496537	0.118292	-2.057177
8	rs10502255	SORL1	0.783217	0.488830	0.025282	0.542160	0.757935	-5.333039
9	rs2243454	SORCS1	0.036963	0.460593	0.025853	0.465575	0.011110	-0.498233
10	rs1945465	GAB2	0.784216	0.487931	0.028139	0.530917	0.756077	-4.298556
11	rs11630227	ADAM10	0.060939	0.465694	0.028710	0.475877	0.032229	-1.018273
12	rs229853	ACE	0.506494	0.560230	0.034138	0.586035	0.472356	-2.580515
13	rs6052810	PRNP	0.410589	0.457041	0.034852	0.479248	0.375737	-2.220671
14	rs12978266	LDLR	0.675325	0.458618	0.038852	0.509934	0.636473	-5.131631
15	rs2637988	IL1B	0.894106	0.441468	0.039566	0.491468	0.854540	-5.000000
16	rs2899664	ADAM10	0.429570	0.456968	0.042565	0.488065	0.387005	-3.109650
17	rs2249483	PRNP	0.267732	0.461420	0.045279	0.481844	0.222453	-2.042363
18	rs2149632	IDE	0.358641	0.457130	0.045993	0.482846	0.312648	-2.571596
19	rs9295802	NEDD9	0.744256	0.491305	0.047136	0.536105	0.697120	-4.480016
20	rs213045	ECE1	0.061938	0.510242	0.047565	0.510391	0.014373	0.000000
21	rs2208842	NEDD9	0.791209	0.448597	0.047850	0.505202	0.743358	-5.660432
22	rs7924472	PICALM	0.232767	0.445043	0.049850	0.465499	0.182917	-2.045563

For most SNPs, there is a great difference in *p*-value, however, the difference in cross validation accuracy is also significant, with the top SNPs of each dataset having higher score in their respective top distribution, meaning that the models trained with different features, are not comparable. However, two SNPs of the SORCS1 gene, rs11193198 and rs2243454, are in the top of both distributions, with cross validation scores of little or insignificant difference between the distributions.

Looking at common the feature importances of the two distributions (figure 4.18 and 4.19) for rs11193198, one sees that the left middle temporal gyrus white matter volume comes up as important in both distributions, while all the other of the common features comes out insignificant. This could indicate rs11193198 being linked to atrophy of this brain area, and could be worth further investigation. In the full feature importance figure for model trained with dataset X shown in the top of figure 4.17, only the left middle temporal gyrus white matter volume comes out as significant. In the feature importance distribution for Datset Y, displayed in the bottom of the same figure, some other features, also from the white matter of the temporal lobes, have been given some slight importance.

For rs2243454, there are less similarities of the common feature importances (figure 4.18 and 4.19), with one feature coming out as slightly significant in the X distribution, getting a negative score in the Y distribution. Few features are given much significance

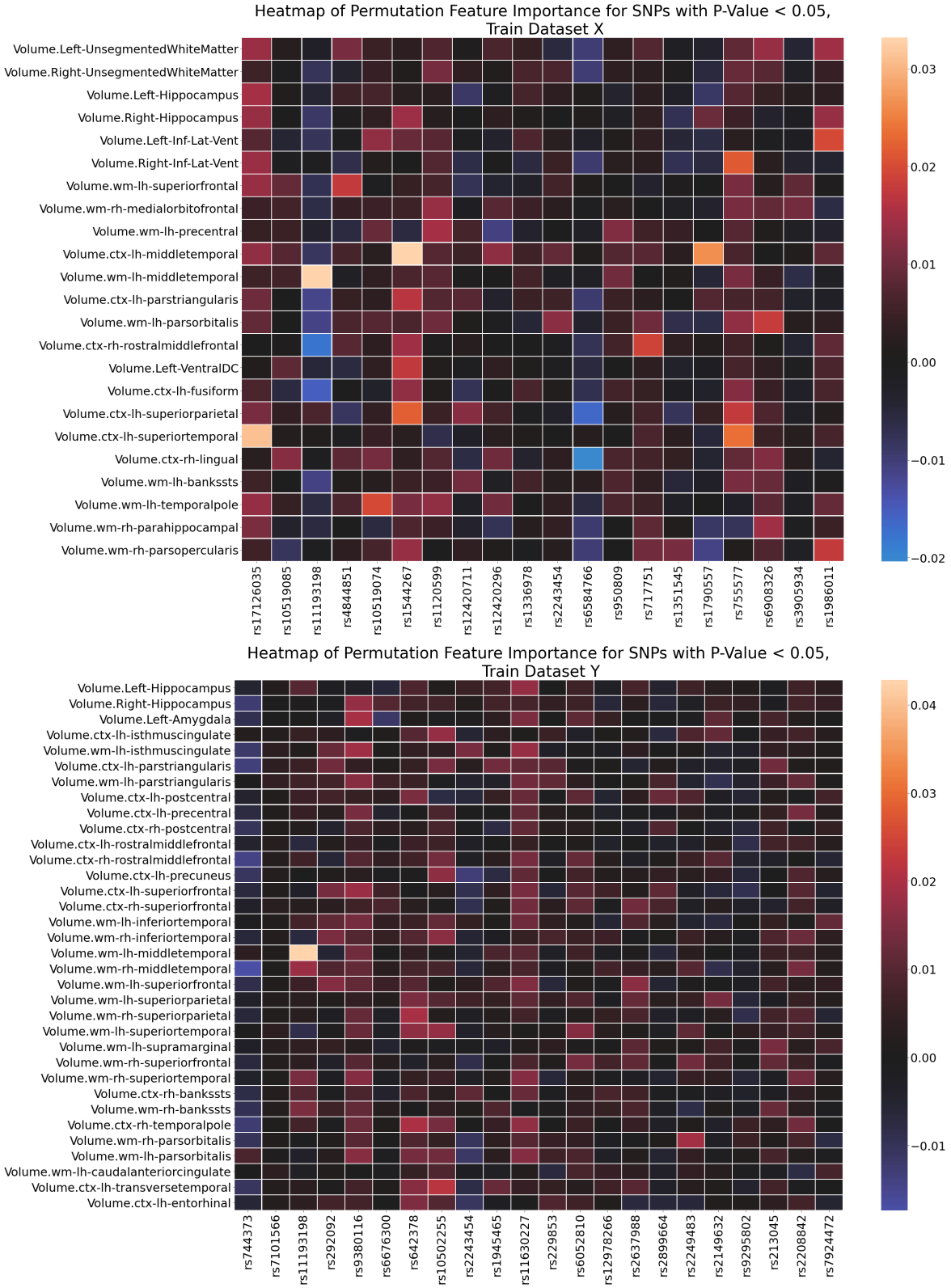


Figure 4.17: Heatmaps of permutation feature importances for top scoring SNPs of their respective distributions.

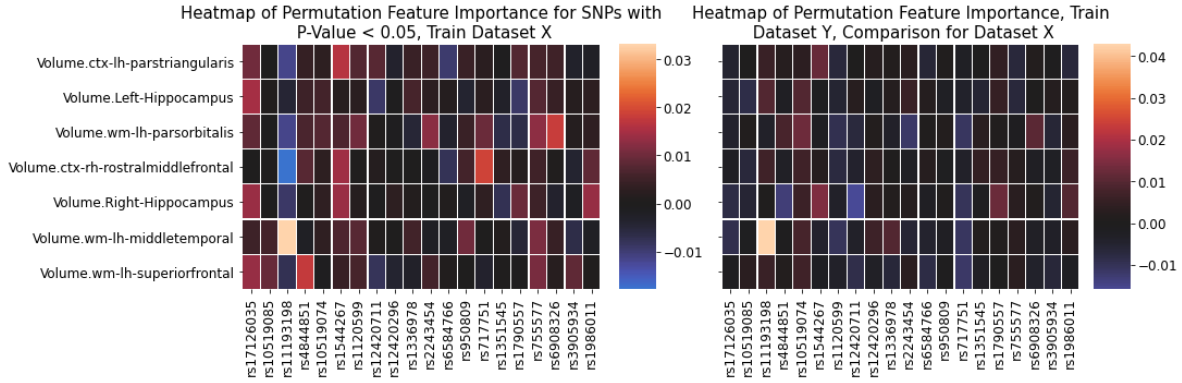


Figure 4.18: Heatmaps of permutation feature importances of common features in Dataset X and Dataset Y for the top scoring SNPs of Dataset X.

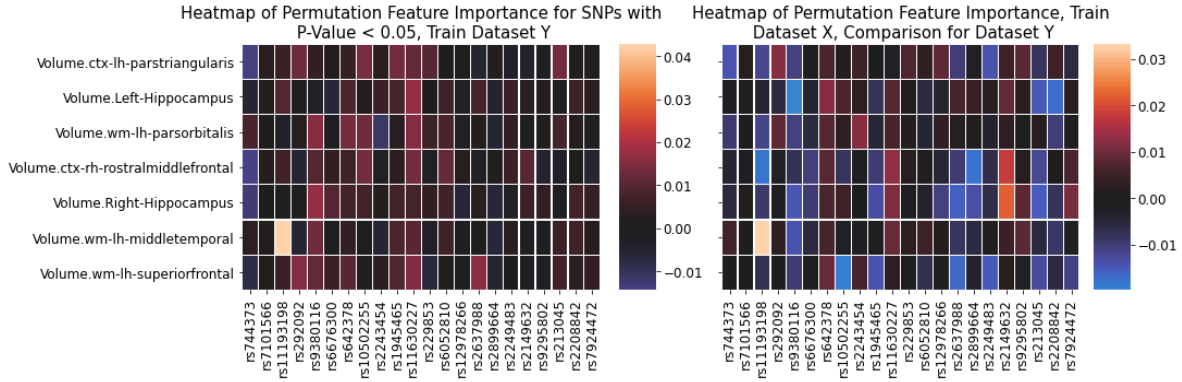


Figure 4.19: Heatmaps of permutation feature importances of common features in Dataset X and Dataset Y for the top scoring SNPs of Dataset Y.

at all, with regard to this SNP (figure 4.17), though this could be due to correlated features being "pushed down" in importance in permutation feature importance testing. In the feature importance for Dataset Y model, some features from the frontal lobe and the cingulate gyri (see figure 2.13 and 2.14) are given some, though very little, importance.

Generally, there are less variation in the feature importances of the Dataset Y model distribution. As this is the dataset with added correlated features, this could be an indication of how the importances of correlated features gets "suppressed" in permutation feature importance measures. This difference in variation is very clear in the comparison of common feature importances for the top scoring SNPs from *p*-value distribution Y in figure 4.19.

4.1.6 APOE ε4 Carrier Status Classification

Selected hyper-parameters for the RFC from the 7-fold cross validation on the train set was 500 estimators and max depth of 5, which achieved cross validation score of 61.99%. This cross validation score is better than the ones achieved in the cross validation hyper-parameter selection in former binary APOE ε4 classification (results in figure 4.6, section 4.1.3), though the two are not exactly comparable, as it was 5-fold

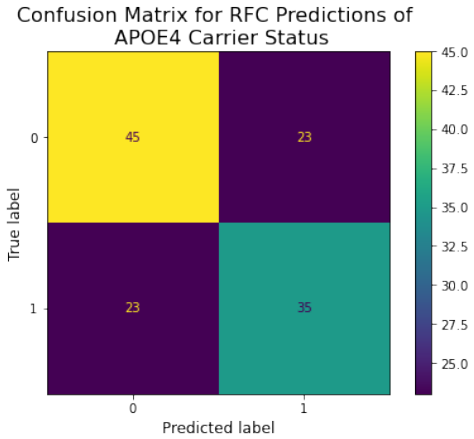


Figure 4.20: Confusion matrix for RF classification of APOE ε4 carrier status using feature Dataset Y.

while this one was 7-fold stratified cross validation, and it is natural that the score should increase with more training data. Contrary, the test accuracy score of this model was 63.49%, which is worse than the former model, which achieved 65.87%. In other words, neither the different features, nor that the new train set is bias corrected, have affected the APOE ε4 classification much.

From the confusion matrix in figure 4.20, accuracy is 66.18% for class label 0 and 60.34% for class label 1. From the permutation tests, the model achieved best possible p-value score of 0.0009990 on the train set. Compared with the permutation test

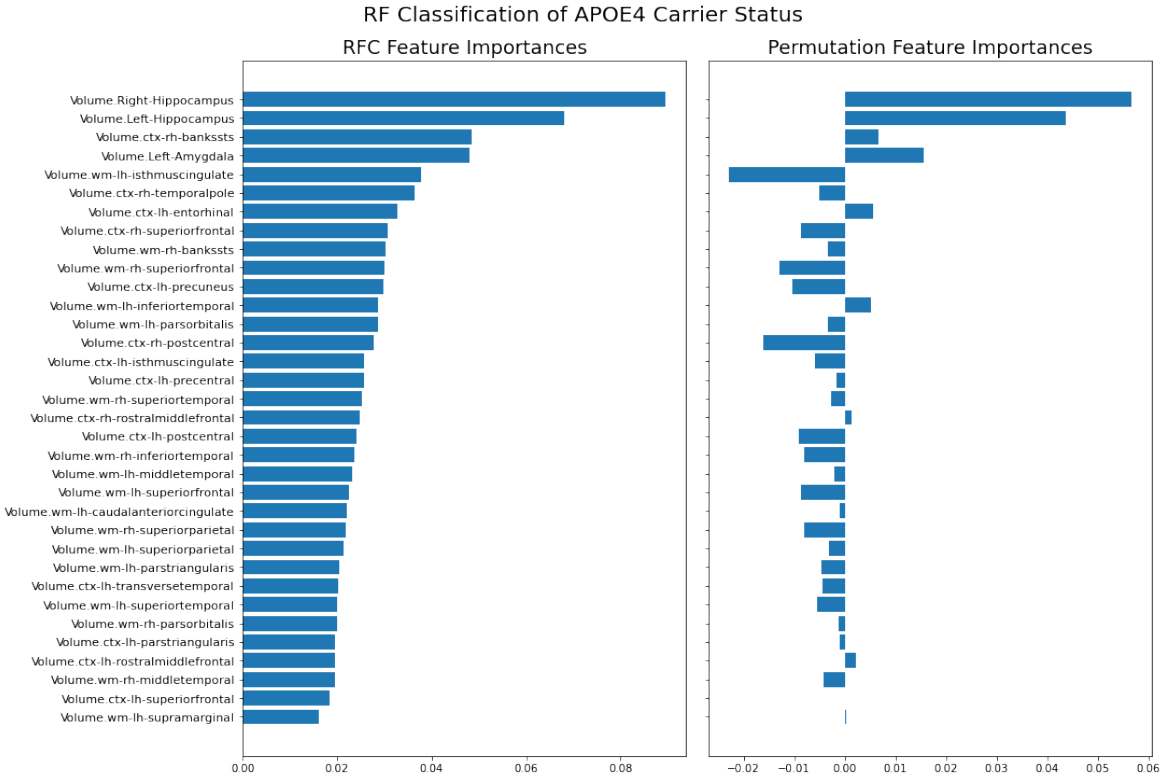


Figure 4.21: Both RF and permutation feature importances for RF classification of APOE ε4 carrier status using feature Dataset Y.

scores in figure 4.2 section 4.1.1, it's clear that feature selection have a great influence on model performance.

The results from the feature importance measures are given in figure 4.21. Both RF importance and permutation importance were given for a comparison between the two. Ideally, they should yield approximately the same results. The top four features remain the same for both, though of slightly different order. That the hippocampus features comes up at top for both, is promising, however, from there on, the difference is great, with the most of the remaining features having negative permutation importance, and the top fifth feature of RF importance, having the greatest negative permutation importance.

4.2 Phase 2: Deep Learning in Imaging Genetics

Table 4.6: DenseNet-121 model descriptions.

Model	Description
Gender	Classification of gender.
AD-1	5-way classification of AD, no pretraining.
AD-2	5-way classification of AD, pretrained with Gender weights.
AD-3	3-way classification of AD, no pretraining.
APOE-1	Binary classification of APOE ε4, no pretraining.
APOE-2	Binary classification of APOE ε4, pretrained with AD-1 weights.
APOE-3	Binary classification of APOE ε4, pretrained with AD-2 weights.
APOE-4	Binary classification of APOE ε4, pretrained with AD-3 weights.
rs11193198	3-way classification of rs11193198, pretrained with AD-1 weights.
rs2243454	3-way classification of rs2243454, pretrained with AD-1 weights.

4.2.1 APOE ε4 Carrier Status Classification with DenseNet

The different DenseNet-121 models will from here on be referred to by abbreviations, explanations of these are given in table 4.6. Figure 4.22 give train loss and validation set accuracy per epoch for each model of the four APOE ε4 training pipelines. From these plots, it can be gathered that the pretrained models start at lower loss than the not pretrained, and they learn faster (train loss drops faster). Validation accuracy also end up higher for the pretrained models, incicating that the not pretrained models need longer training time (more epochs) to reach the same results. The plots also indicate that the best pretraining pipeline for APOE ε4 is the APOE-2 model, pretrained with AD-1 weights, namely the 5-way classification of AD with no pretraining. This model has the fastest dropping loss, and achieves the highest validation accuracy. Interestingly, the validation accuracy plot for APOE-4 (pretrained with 3-way classification of AD) fluctuates more than the other APOE models.

The confusion matrices of these models, given in figure 4.23, support the observation that the APOE-2 model is the best. Though none of the models do particularly well,

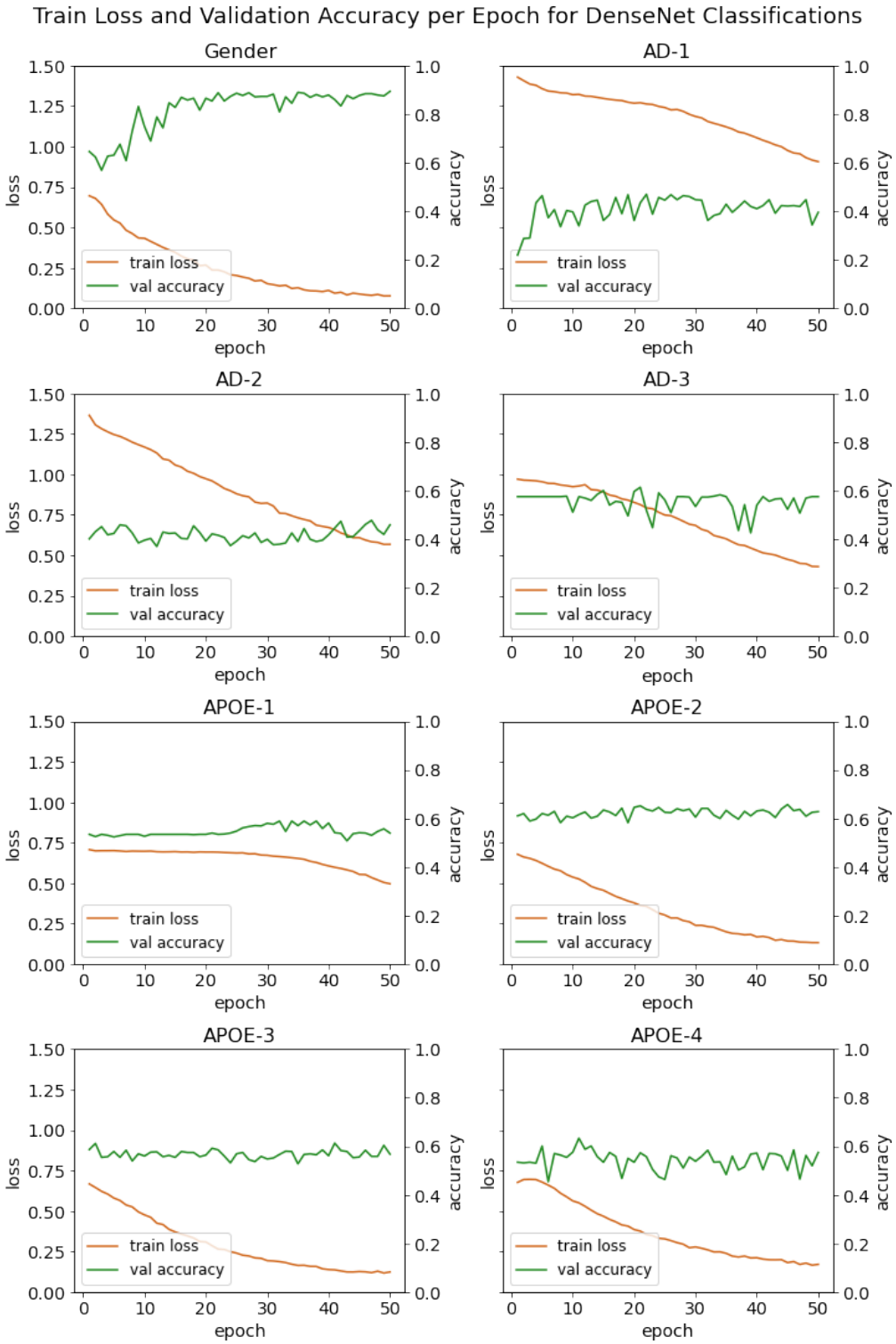


Figure 4.22: Train loss and validation accuracy per epoch for all models of the APOE ε4 pretraining pipelines. (Read from left) first row: Gender model, AD-1 model; second row: AD-2 model, AD-3 model; third row: APOE-1 model, APOE-2 model; fourth row: APOE-3, APOE-4. The left axes of the graphs give loss, while the right axes give accuracy.

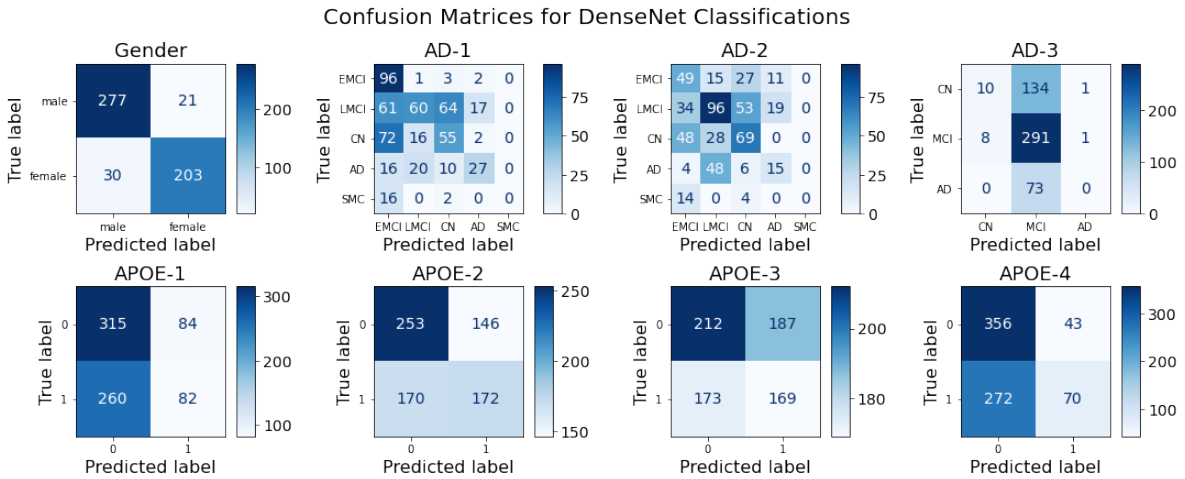


Figure 4.23: Confusion matrices for the DenseNet-121 models from the four pretraining pipelines.

this model seem best at distinguishing the two classes. The APOE-1 and APOE-4 models are heavily favoured to the majority class, and the APOE-3 model seem almost to predict arbitrarily. The superiority of the APOE-2 model is confirmed by its F1 score, and other performance measures in table 4.7. As the majority class representation is 53.85%, the APOE-1 and APOE-3 models come out rather terribly with accuracy scores lower than the majority class representation. The APOE-1 model would most likely have benefited from longer training (more epochs).

The confusion matrices for the AD models give a possible explanation for the terrible performance of the APOE-3 and APOE-4 models. Though the AD-3 model has the highest accuracy test score, this is due to it simply predicting the majority class. The few predictions of the other two classes is possibly completely due to chance. The only indication to the contrary, is that no "CN" (healthy) subject has been predicted as "AD" or vice versa. Merging the "LMCI" and "EMCI" classes was evidently not a good idea. Most likely due to the differences between the two classes being greater than anticipated, and, at the same time, the similarities between the "AD" and "LMCI", and the "CN" and "EMCI", classes being greater than expected too. Though the AD-1 and AD-2 models are not able to distinguish the minority class, "SMC", it is either predicted as "EMCI" or "CN" which is the two classes that should be the most similar to it. The two models seem also to do well in separating the "AD" and "CN" classes. The difference between "LMCI" and "AD", and "CN", "EMCI" and "LMCI" is clearly a greater struggle. The AD-1 has managed to distinguish that "EMCI" is the class with most in common with all classes, and has therefore managed to predict almost all instances of this label, and achieves a greater accuracy score than AD-2 (see table 4.7). The AD-2 model is seemingly more favoured to the majority class - "LMCI", and does therefore have much lower accuracy for the "AD" and "EMCI" classes compared to AD-1.

The performance measures (table 4.7) and confusion matrix (figure 4.23) for the Gender model, is a good illustration of how well a single 3D CNN can perform on a brain image classification task.

Table 4.7: Test scores for the DenseNet121 classification models. For model description, see table 4.6. For binary classification, precision, recall and F1 scores are listed in addition to accuracy.

Model	Accuracy	Precision	Recall	F1 score
Gender	90.40%	90.63%	87.12%	88.84%
AD-1	44.07%			
AD-2	42.41%			
AD-3	58.11%			
APOE-1	53.58%	49.40%	23.98%	32.28%
APOE-2	57.35%	54.09%	50.29%	52.12%
APOE-3	51.42%	47.47%	49.42%	48.42%
APOE-4	57.49%	61.95%	20.47%	30.77%
rs11193198	47.65%			
rs2243454	37.57%			

4.2.2 3D DenseNet Classification of rs11193198 and rs2243454

The SNPs rs11193198 and rs2243454 were selected for DL classification, due to them coming up in the top of both *p*-value distributions from section 4.1.5.2. In particular, rs11193198 is of special interest since it had common features of importance in both distributions. From the above results, the pretraining scheme with the weights of the AD-1 model was selected for the rs11193198 and rs2243454 models. The train loss and validation accuracy per epoch plots given in 4.24. They are similar, with little or no increase in validation accuracy, though train loss drops significantly.

Rs11193198 has a minority class, the "0" label, while the other two classes are quite evenly distributed (45%). From the confusion matrix, it is clearly favoured towards class "1", while the predictions of label "0" are seemingly arbitrarily. Looking at the data of the predictions of this label, they seem even more random, as the four true predictions belong to different subjects. However, on average, the model predicts the same label for 77.82% images of the same subject, which might indicate that the model is not making its predictions completely at random.

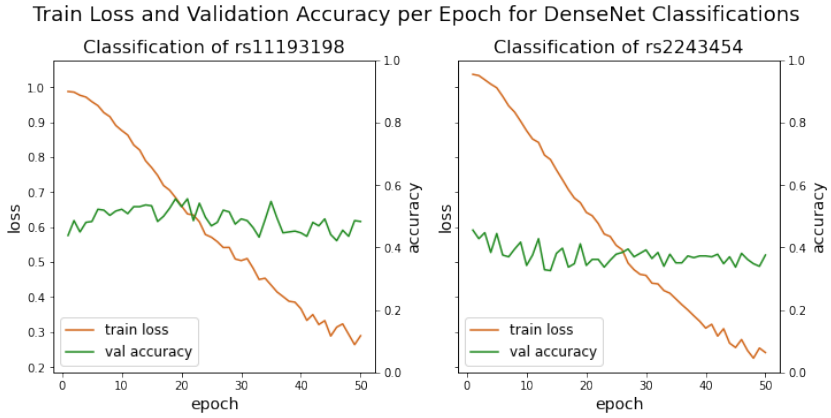


Figure 4.24: Train loss and validation accuracy per epoch for the two SNP model pipelines. From left: rs11193198 model, rs2243454 model.

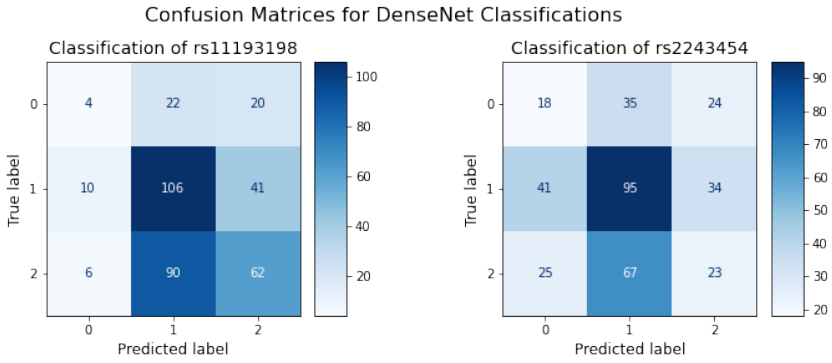


Figure 4.25: Confusion matrices for the two SNP model pipelines. From left: rs11193198 model, rs2243454 model.

On the other hand, rs2243454 has a majority class, the label "1", represented by approximately 48% of the dataset. The accuracy test score, given in table 4.7, for rs2243454 is worse than its majority class representation. The confusion matrix of rs2243454 in figure 4.25, shows that it clearly favours the majority class, while no information can be gathered on how it predicts the other two classes, it is seemingly random. However, looking directly at the prediction data, the general trend is that the model predicts mostly the same label for images of the same subject. On average, 76.79% of images of the same subject are given the same label.

4.2.3 Occlusion Sensitivity

The AD and APOE models selected for computing occlusion sensitivities were AD-1 and APOE-2, based on their results. All occlusion sensitivity heatmaps can be found in Appendix 2. Only a few maps will be presented here as well, and the rest will be referred to their figure label in the appendix.

Figure 4.26 displays the effects of mask and stride size on occlusion sensitivity on these images. Mask size of $16 \times 16 \times 16$ and stride size of $4 \times 4 \times 4$ have been selected as the most informative coupling, having a big enough mask size to significantly affect the results, and at the same time, a small enough stride to highlight the most important areas. The smaller mask size of 8 does not have enough impact on the test results, i.e. the model still gain the information necessary for its predictions from the neighbouring datapoints. On the other hand, the bigger mask sizes of 24 and 32, occludes too much, and we no longer gain any information on which areas of the brain are more important

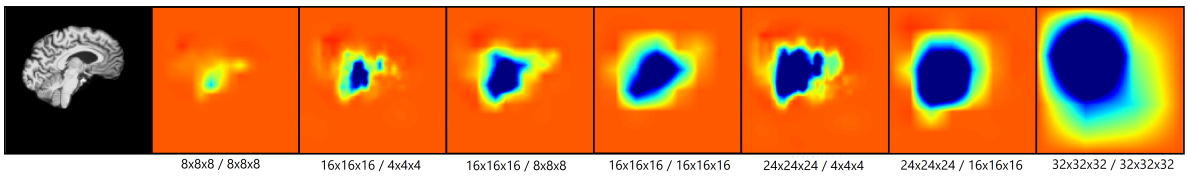


Figure 4.26: The effect of mask size and stride size on occlusion sensitivity. To the left is the brain image slice of the occlusion maps. The mask size and stride size coupling will be given as a/b, where a gives the $a \times a \times a$ mask size, and b gives the $b \times b \times b$ stride size. From the left: 8/8, 16/4, 16/8, 16/16, 24/4, 24/16, 32/32.

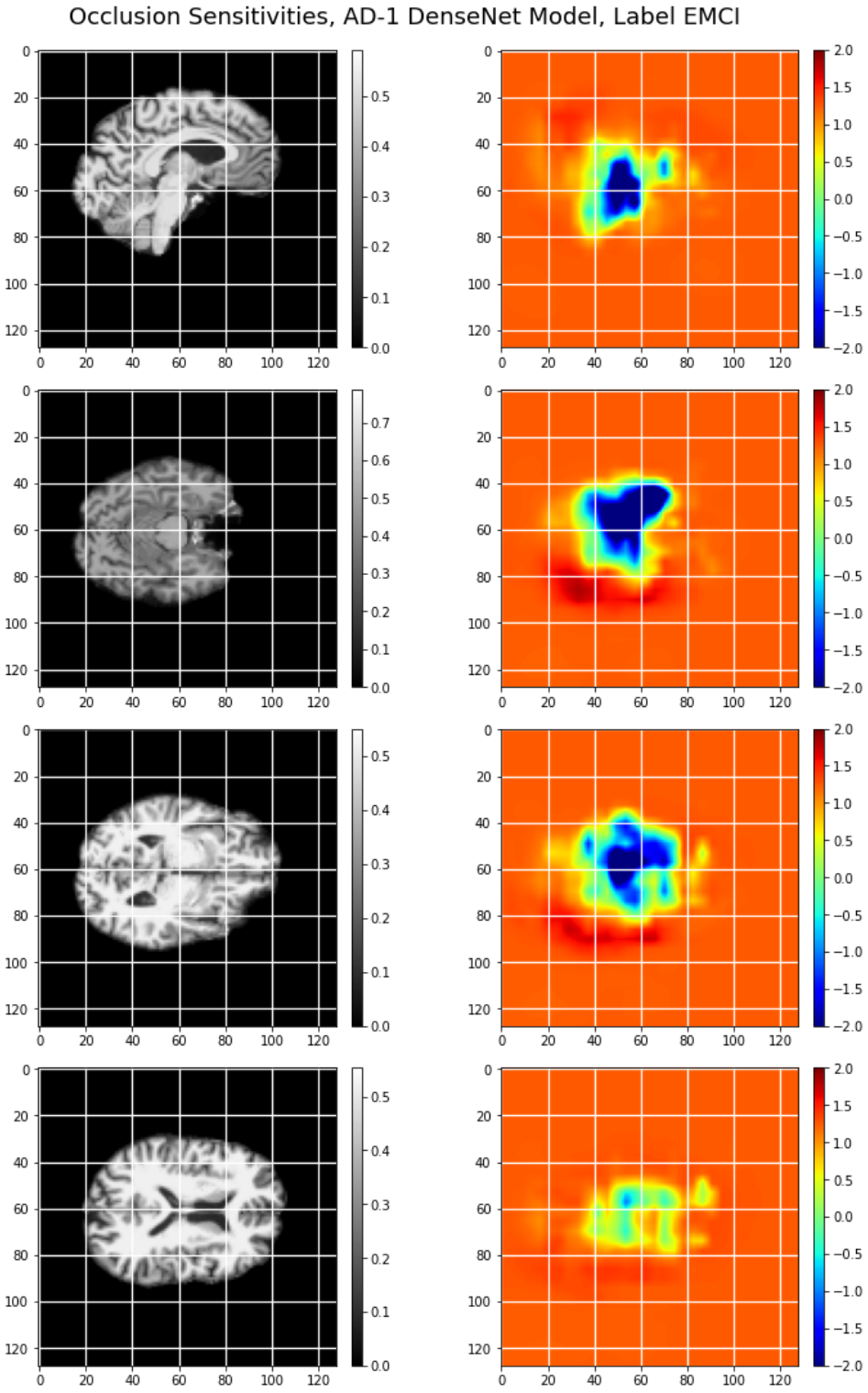


Figure 4.27: Occlusion sensitivity maps for AD-1 DenseNet model (5-way classification of AD, no pretraining), for prediction of label "EMCI". To the left is the image slice, to the right is the occlusion map of this image slice. Slices have been made along the following axis (from top): x=60, y=60, y=50 and y=40. For the definition of the axes, see figure A.2.1 of Appendix 2.

for the prediction. The same can be said for the bigger stride sizes of 8 and 16, - the areas of importance become too big to be informative.

As the images are 3D, they must be sliced along a plane for heatmap illustrations. How the axes have been defined is explained in figure A.2.1 of Appendix 2. However, shortly put, thinking of a person standing, the x axis have been defined as going horizontally, in direction from left shoulder to right shoulder, the y axis as going vertically, from top to bottom, and the z axis as going horizontally in the direction a person face. Slicing at $x = 60$, means slicing along the yz-plane at $x = 60$.

Blue areas indicate areas of importance for the prediction, i.e. areas that when occluded, caused a drop in prediction probability. Green to yellow areas indicate neutral areas, while yellow to red areas give areas that when occluded, increases the prediction probability, i.e. that confuses the model. Figure 4.27 display occlusion sensitivity heatmaps for the AD-1 model. First, a slice trough approximately the middle of the brain, between the two cerebral hemispheres, is displayed. The areas of interest from this slice are then further investigated by slicing along the horizontal (xz) plane. From the first slice, areas of or around the brain stem, and possibly parts of the cerebellum, are important for the prediction. The slices along the horizontal plane confirms this view, and possibly indicates that the left temporal lobe is of greater importance than the right. These areas of importance surrounding the brain stem are in perfect position for the hippocampus, other structures of the limbic system and hypothalamus and other structures of the diencephalon. The occlusion heatmaps of other AD-1 classes, are found in figures A.2.3-A.2.7 in Appendix 2. Occlusion maps for the "SMC" label is given in figure A.2.3. Here, "SMC" is not the predicted label, and the map does therefore seem "opposite" of the others. Occluding all areas decrease the prediction of "SMC", particularly the temporal lobes. Figure A.2.7 show the occlusion heatmaps for label "AD". For this prediction, the frontal lobes and cerebellum are important for the prediction. Considering atrophy of these brain structures generally occurs at later stages than that of the temporal lobes, these results makes sense.

Occlusion sensitivity heatmaps for the APOE-2 model are displayed in figure A.2.8 and A.2.9 of Appendix 2. Figure A.2.8 is for prediction of label "0", for an image of this label, while A.2.9 is for label "1". From figure A.2.8, it is clear that the brain stem and structures surrounding it have been important for the prediction of "0", while for prediction of label "1", from figure A.2.9, structures more to the front of the brain, possibly of the cerebral cortex of the temporal or frontal lobes are important for the prediction. This is approximately the same way that the AD-1 model separates the "AD" and "EMCI" classes (figures 4.27 and A.2.7.

From the results of the permutation feature importance tests (figure 4.17) of the two RFC model classifications of rs11193198 from section 4.1.5.2, white matter structures of the temporal lobes came out as important for rs11193198 classification. These structures also come out in the occlusion sensitivity maps for the rs11193198 model in figures A.2.10-A.2.13 of Appendix 2. The two bottom heatmaps from figure A.2.13 highlight the left temporal lobe in both the xz- and xy-planes. The other heatmaps are more difficult to interpret, though the white matter structures of the left occipital

and temporal lobes are possibly highlighted in the second to bottom heatmap of figure A.2.10, the middle and bottom heatmap of figure A.2.11 and the bottom heatmap of figure A.2.12. The left cerebellum is shown as very significant in the bottom heatmap of figure A.2.10 and middle heatmap of figure A.2.11. The only features from the occitpital lobe in the RF classifications are the gray matter of the right lingual gyrus and the gray matter of left fusiform gyrus (which spans the occipital and temporal lobes) of Dataset X, neither of which are given any permutation importance for rs11193198 classification.

The permutation feature importances (figure 4.17) of the two RFC model classifications of rs2243454 from section 4.1.5.2, were not particularly informative, with all features from Dataset X having neutral to weakly positive importance, and most features of Dataset Y having weakly negative, neutral or weakly positive importance. Only one feature of Dataset X, the white matter of the orbital part of the left inferior frontal gyrus, and one feature of Dataset Y, the isthmus of left cingulate gyrus (white matter), part of the limbic system, were given somewhat importance. Figures A.2.14-A.2.16 of Appendix 2 displays the occlusion sensitivity heatmaps of the rs2243454 model. Interestingly, for the majority class label "1", no areas are highlighted (figure A.2.15). This could possibly indicate, that when the DenseNet model see no "signs" of the other two classes, it simply predicts the majority class. In predicting label "0" (figure A.2.14), the left cerebral hemisphere (frontal, temporal and possibly parietal lobes) and right cerebellum are highlighted as important. For the prediction of label "2" (figure A.2.16), the superior parietal lobules of the parietal lobes and the cerebellum are highlighted.

Figure A.2.17 and A.2.18 of Appendix 2 show the occlusion sensitivity maps for the high performing Gender model. The greatest difference between these heatmaps, and the ones for the other (weak) models is that they are more "concentrated". Only one area of each of the two labels are marked as important; a smaller area centred around (55,70,70) on the left temporal lobe for label "male" (figure A.2.17) and a bigger, more diffuse area on the left cerebral hemisphere around the parietal and frontal lobes.

Ideally, the background of the brain images should have been green in the heatmaps. That would have signified that they had no impact on the predictions. This is, however not the case. At best the backgrounds are marked orange, at worst dark red/blue. "Extracting" the brain from the images, i.e. cropping such that as much background as possible is removed, could yield a significant increase in classification performance.

5 Discussion and Concluding Remarks

5.1 Phase 1: Random Forest Classifier in Imaging Genetics

As mentioned in section 2.7.3, established imaging genetic approaches, such as univariate, multivariate and voxel-wise approaches are prone to disadvantages such as being computationally expensive, selection of IPs for such computationally expensive methods taking time and requiring knowledge of which features to choose, and/or relationships between different IPs and genetic markers being lost. The RFC approach in this thesis were meant to tackle most of these issues, the exception being the loss of relationship between genetic markers. RFC models are fast to train, even on high-dimensional data, and so the effect of multiple IPs can be tested at the same time, without great loss in speed. ML feature selection methods circumvent the issues of hand-picking IPs. However, with regard to performing successful and efficient genome-wide studies, these RFC experiments have come out rather unsuccessful.

Table 5.1 displays results from the GWAS1 permutation tests p-value distributions of SNPs previously reported in literature to be associated with hippocampal, amygdala and/or cortical atrophy. Though the goal of these experiments has not been to recreate results from previous studies, ideally, at least some of these SNPs, particularly those associated with hippocampal volume, as that have been a feature present in all feature selected datasets, should have performed well in these studies. Not only does none meet the threshold of 0.05, most are high above. There were however, many weaknesses with this approach. One weakness was, as discussed before, the RF importance based feature selection. Another was that the features were not bias corrected, which was addressed for GWAS2. Uncorrected IPs showed a strong systematic difference between gender, as observed in the relative difference between the distributions in figures 4.12b and 4.13b. A third weakness was that the number of permutations run was only 50 for

Table 5.1: P-value results from stage 1 and 2 of the third p-value estimation through permutation testing from GWAS1 for SNPs associated with hippocampal, amygdala and/or cortical volume [12; 54].

SNP	Gene	GWAS1-3.1 (p-value)	GWAS1-3.2 (p-value)
rs7561528	BIN1	0.078431	0.346535
rs610932	MS4A6A	0.156863	
rs2075650	TOMM40	0.235294	
rs6896317	TRIO	0.117647	
rs439401	APOE	0.411765	
rs405509	APOE	0.509804	

Table 5.2: P-value results from GWAS1 and GWAS2 of SNPs listed as the polymorphism of the AlzGene Database top 42 genetic risk factors, that exist in the datasets. GWAS1-3.1 and GWAS1-3.2 refer to stage 1 and 2 of the third p-value estimation through permutation testing from GWAS1. GWAS2-X and GWAS2-Y refer to the GWAS2 p-value distribution X and Y, respectively.

SNP	Gene	GWAS1-3.1 (p-value)	GWAS1-3.2 (p-value)	GWAS2-X (p-value)	GWAS2-Y (p-value)
rs11136000	CLU	0.333333		0.389610	0.276532
rs3851179	PICALM	0.196078		0.519481	0.125696
rs744373	BIN1	0.450980		0.058941	0.001286
rs5930	LDLR	0.980392		0.279720	0.755035
rs4878104	DAPK1	0.509804		0.502498	0.609627
rs5984894	PCDH11X	0.352941		0.558442	0.539780
rs1801133	MTHFR	0.843137		0.134865	0.817883
rs1143634	IL1B	0.137255		0.281718	0.322954
rs939348	THRA	0.960784		0.708292	0.216969
rs2306604	TFAM	0.176471		0.485514	0.704185
rs213045	ECE1	0.058824	0.039604	0.061938	0.047565

the first stage, and 100 for the second stage. For reliability and computing small enough p-values, permutation tests must be run for a significant amount of permutations [63]. Using the balanced binary dataset example of 100 datapoints, given in 2.8.2, which has $\binom{100}{50} \propto 10^{29}$ possible permutations, let the null hypothesis not be rejected for 10,000 of these permutations, that will still yield a p-value of $(10,000+1)/10^{25} \propto 10^{-29}$. If extremely unfortunate when performing tests for 100 permutations of this dataset, most of them could belong to this set of 10,000 permutations. However, this is statistically very little likely ($p \approx 10^{-23}$). Table 5.1 does however, have one SNP that has performed significantly worse for 100 permutations than for 50, while table 5.2 presents one SNP which achieves better p-value for 100 permutations compared to 50.

Table 5.2 list the p-value results from the different genome-wide studies performed in phase 1, of SNPs from the AlzGene Database top genetic risk factors[1]. Though all SNPs of this table have not necessarily been associated with brain atrophy directly, association with AD is still established. Yet, only two of these SNPs have p-values below the threshold of 0.05; rs744373 in GWAS2-Y, and rs213045 in GWAS1-3.2 and GWAS2-Y. Rs744373 have formerly been found associated with amyloid deposition [12], while one polymorphism of rs213045 is associated with decreased risk of AD, possibly through endothelin-converting enzyme-1(ECE-1) being a candidate Aβ-degrading enzyme in brain [56].

One of the top 20 SNPs from the GWAS2-X distribution, in table 4.4 in section 4.1.5.2, have previously been found associated with AD in the Chinese Han population, rs950809 [123]. Apart from rs744373 and rs213045 from the top SNPs from the GWAS2-Y distribution in table 4.4, rs2149632 have previously been found associated with AD [79].

[1] https://web.archive.org/web/20110222233234/http://www.alzgene.org/TopResults.asp

In order to perform better, chances are that 1. the RFC model require more data to learn and 2. accuracy simply is not a good choice of performance measure with regard to null hypothesis testing for such unbalanced datasets. With balanced datasets of the same size for all SNPs, chances are that RFC model accuracy could have worked well as test statistic for the GWAS experiments. However, by limiting datasets to the minority class, very few SNPs would have large enough datasets for training. One of the filtering criteria from section 3.3.5.1 was that the minority class (for 3-class SNPs) should be represented by at least 5% of the datapoints. For, at best, 628 subjects, this is 32 datapoints. In most cases with highly heterogeneous, high-dimensional data, a 3-class dataset of 96 datapoints will be insufficient for training ML algorithms. In the same way, it can be argued that 32 datapoints is insufficient for an RFC to learn the class structure in the current tests, i.e. that the 5% representation criteria is too low.

With regard to efficiency, another method for computing p-values than permutation tests is necessary. In Huang et al.'s FVGWAS framework, they introduced a wild bootstrap detection procedure for a heteroscedastic linear model [54]. Adapting this procedure to create a RFC GWAS framework, could be worth investigating.

This thesis have focused on feature selection for limiting the dimension of the dataset, however, there is a great chance that RFC models would have performed better with features made with feature extraction methods, such as PCA. PCA, independent component analysis (ICA) and independent vector analysis (IVA) are common tools in multivariate analysis in imaging genetics, used both with reducing dimension of brain-wide data and in sets of genotype analysis [72]. However, model interpretation would then have been much more complex, and a great interest in this study was to see which brain structures the SNPs affected. Methods such as PCA loadings can yield how much the original features contributed to the principal components (made features), but if the contribution is from a great set of features, this is still not very informative. A possible solution could be to perform permutation importance where the original features are permuted before transformation, and then prediction is performed with the transformed features.

5.2 Phase 2: Deep Learning in Imaging Genetics

In 2020 Ruiz et al. performed a 4-way classification of AD diagnosis with an ensemble of 3D DenseNets on 3D MRI brain scans [98]. The diagnostic labels were "CN", "EMCI", "LMCI" and "AD". This ensemble, consisting of three DenseNet-121 models of different growth rates and convolutional kernel sizes, achieved test accuracy of 83.33%. Predictions were made using a probability based fusion of the individual models' results, rather than majority vote. The individual accuracy scores for the three models are given in table 5.3. This study used 600 preprocessed MRI scans in NIfTI format from ADNI2 and ADNIGO, divided into 80% train and 20% test sets. Both train and test sets were balanced. For the DenseNet-121 classifications in this thesis, the same hyper-parameter settings as for these models have been used, except a growth rate of 32, convolutional kernel size of $7 \times 7 \times 7$ and training for 50 epochs (Ruiz et al. trained for 100 epochs, and used no pretraining). Adding that we performed a 5-way

Table 5.3: Test performance of the three individual models of the 3D DenseNet ensemble for 4-way classification of AD diagnosis from study performed by Ruiz et al. in 2020 [98].

Model nr.	Growth rate	Convulutional kernel size	Accuracy
1	32	$3 \times 3 \times 3$	53.33%
2	22	$7 \times 7 \times 7$	57.50%
3	28	$7 \times 7 \times 7$	66.67%

(CN"/"EMCI"/"LMCI"/"AD"/"SMC") and 3-way (CN"/"MCI"/"AD") classification of AD instead of 4-way, none of these results are directly comparable. The accuracy test performance of the AD-1 model (44.07%) does however seem meagre in comparison with what's possible for a single 3D DenseNet-121 model in classification of AD from table 5.3. However, Ruiz et al. used a balanced test set, thus their model performance does not reflect performance on real-life data (generalization). Real-life data from the general population will most definitely not be balanced. Hence, the AD-1 model test performance is more realistic and reflecting of the real world in comparison.

Interestingly, it is the models 2 and 3 from 5.3, which have much fewer parameters than model 1, that achieves better results. Another perk of these two models is that fewer parameters generally mean shorter training time. Ruiz et al. also used a much smaller, but balanced, train dataset. Using balanced train data could be an easy way to increase model performance. However, the limited amount of MRI data available, especially with regard to classification of SNPs, could pose a problem to this strategy. Then, the problem is not just that there is generally a limited amount of available MRI and genetic/diagnostic data, but that the minority class will cause a great restriction on the size of the dataset. Still, Ruiz et al. have used only 600 images with success.

What really boosted the performance of the 3-way classification performed by Ruiz et al., however, was the use of the ensemble technique. This is not the first time ensembles of DenseNets have been used in AD classification. Wang et al. performed three binary classifications ("MCI"/"AD", "CN"/"AD" and "CN"/"MCI") and one 3-way ("CN"/"MCI"/"AD") classification of AD using an ensemble of 3D DenseNets [121], the 3-way classification achieving an impressive performance accuracy of 97.52%. This article also performs a thorough testing of the effect of the different DenseNet hyperparameters on AD classification. Many other techniques have been used in CNN classification of AD, achieving good results [1; 14; 52; 71; 100; 115], however, the goal of this project has not been achieving the best possible classification score of AD. Classification of AD is simply used as a pretraining scheme. The problem of these well-performing CNN models is that they are often complex, require long training time and/or are difficult to interpret. The goal of imaging genetics is to assess the effect of SNPs (or other genetic variants) on body function. Therefore, interpretability of the model is more important than achieving the best possible classification performance. On the other hand, model performance must be good enough to be considered reliable. This discussion will therefore focus on (hopefully) efficient methods for boosting the performance of single 3D CNN models.

Image preprocessing can greatly affect CNN model performance, e.g. the background

of the MRI brain scans affect performance, as seen in the occlusion sensitivity maps in Appendix 2. So, as mentioned before, cropping the brains out of the images, then scaling them to the same size, could be a good measure to increase model performance. Another preprocessing method is spatial normalization, making each image voxel of every image correspond to the same anatomical position. There exists many softwares for spatial normalization, such as SPM12 [13]. Spatial normalization was one of the preprocessing steps Tang et al. used on the MRI scans in their presentation of a CNN for Computer Aided Diagnosis, named 3D fine-tuning convolutional neural network (3D-FCNN), in 2018 [115]. This model was made to balance speed and accuracy, and achieved an accuracy of 91.32% in 3-way classification of AD ("CN"/"MCI"/"AD") using 3744 MRI scans from 321 subjects from the ADNI database. Another preprocessing step they employed was Gaussian kernel smoothing of the 4D image (all 3D images across subjects). Performing SNP classification with these preprocessing steps, as well as the proposed 3D-FCNN model, could be worth investigating.

The great performance of the Gender model (section 4.2.1) indicate that a predominant signal in the MRI brain scans is gender. 3D CNNs have also been used with great success in brain age regression [20; 59]. Bias correction for age and gender had impact on the RFC analysis with regard to feature selection (AD classification) and the SNP classifications, though not on APOE $\varepsilon 4$ classification. Confounding correction could therefore help uncover more subtle SNPs or AD related effects. This gender bias could also be the reason why the AD-2 model performs worse than AD-1; that starting off training with weights for gender classification increases the gender bias. Logically, That bias correction should help increase performance also makes sense logically. Brain atrophy is a normal part of aging, just not as extreme as the brain volume decreases of AD and MCI patients. Therefore, consider the classes "CN" and "EMCI" from the AD classifications, will the normal brain volume decrement of an elder "CN" be distinguishable from that of a younger "EMCI"?

In 2020, Dinga et al. stated that the most common way to control for confounds in neuroimaging, namely adjusting voxels for confounds using linear regression before training (like we did with the RFC data), is not sufficient for ML algorithms, as they can extract information about confounds that regression does not remove [24]. As an alternative, they proposed a method to estimate the effect of confounds on the output, rather than correcting the input, by simply applying the traditional regression techniques for confound adjustment on the ML model output.

Another method for confounding correction, where alterations to the DL model architecture itself is performed, is the confounder-free neural network (CF-Net) proposed by Zhao et al. in 2020 [126]. This model architecture for confounder-free learning is composed of three subnetworks. Two of these subnetworks are the subnetworks of standard CNNs: a feature extractor consisting of convolutions and pooling layers, followed by a predictor, commonly of fully connected layers. The last subnetwork is a predictor of the confounder which guides the feature extractor in removing confounding effects through the min-max game as done by generative adversarial networks (GANs) [126].

In these experiments, CNN classification of gender has been used for pretraining, but

what if there instead was some way to use it to subtract the gender bias? In 2019, Wang et al. proposed such an approach for confounding correction that they named *Confounder Filtering* (CF) [122]. The procedure is to 1. train the DL model on the task at hand, 2. replace the top model layer with one predicting the labels of confounding factors, 3. start training again, keeping track of the weights updates, 4. replace all weights that are frequently updated during this training phase with zeros. Wang et al. improved performance of four DL models (lung adenocarcinoma prediction, heart right ventricle segmentation, brain tumour prediction and student's confusion status) by applying CF to the models [122]. However, if many classifications are planned for the same dataset (like in this study), it would be advantageous if the confounding prediction and classification tasks could be performed separately, i.e. training individual models of the same architecture on prediction of confounding factors and on the classification tasks, and then the frequently updated weights from these confounding models are replaced with zeros in the others. This would be a more efficient approach when performing many classifications for the same dataset, and could be worth investigating.

In 2019, Pominova et al. proposed a method to avoid extensive preprocessing of 3D MRI for CNNs, namely 3D deformable convolutions (d-convolutions) [90]. 3D MRI brain scans are noisy and high-dimensional, and traditional CNNs are very sensitive to image size, scale and spatial orientation [90]. 3D d-convolutions is proposed as a method to make the network itself invariant to these variations in the input images. Pomina et al. achieved significant increase in ROC/AUC performance with d-convolutional VoxResNet model with comparison to normal VoxResNet model in classification of Schizoprenia and Bipolar Disorder.

The pretraining scheme chosen for the rs11193198 and rs2243454 models were AD-1 weights. Another option could have been pretraining with weights from one of the APOE models. This classification task is after all more similar to SNP classifications than AD diagnosis. With the APOE-2 model weights, this could also have tested the effect of a double pretraining scheme without the (possibly) extra gender bias.

As CNN models are time consuming to train, especially with 3D images, they are (at present) not suitable for GWAS studies like in the RFC phase of this thesis. However, they could provide a powerful tool in assessing the impact of single SNPs on brain (or other body) function, like the experiments with rs11193198 and rs2243454. The occlusion sensitivity heatmaps are by far more easy to interpret than the feature importance of the manually extracted features from the RF classifications. However, occlusion sensitivity comes with several drawbacks; computing occlusion sensitivity maps is highly time consuming (depending on stride size), and is individual for each image and prediction. Ideally, one should be able to easily visualize the generally important brain areas for predicting a label, not for one single image and prediction. Further research in DL model explanation visualization is therefore necessary. One approach could be computing the average of occlusion sensitivity maps for several images. Naturally, this would only work for spatially normalized images. However, this would be highly time consuming.

Alternative visual explanation techniques include Guided Backpropagation [110] and

Deconvolution [125]. Deconvolution map the feature activity back to the input pixel space using a Deconvolutional Network (deconvnet). A deconvnet can be thought of as a reverse CNN, mapping labels to pixels using the same components as a CNN [125]. Guided Backpropagation is a variant of Deconvolution, where backward flow of negative gradients are prevented [110]. These methods are, however, not class-discriminative [104]. In 2017, Selvaraju et al. presented an alternative to occlusion sensitivity named Gradient-weighted Class Activation Mapping (Grad-CAM) [104]. Grad-CAM highlights the important image regions for predicting the given label by using the gradients of the label flowing into the final convolutional layer. Results of this technique highly correlates with occlusion sensitivity, but requires only a single forward pass and partial backward pass per image, and is thus typically an order of magnitude more efficient [104].

5.3 Conclusion

With regard to performing successful and efficient genome-wide brain-wide studies, the RFC experiments in this thesis have been ineffectual. The models have yielded relatively low performance, and finding effectual ways to compare the accuracies between SNPs have posed difficulties. Gender was found to have strong effect on uncorrected features for the RF classifications, and to be a predominant signal in the MRI brain scans with regard to the CNN classifications. Correcting for confounding factors such as gender and age is therefore necessary for these experiments. Though the DenseNet model classifications of SNPs rs11193198 and rs2243454 yielded low accuracies (47.65% and 37.57%), CNNs could still be a possible powerful tool in assessing the effect of SNPs on the brain. A larger cohort is desirable with regard to both RF and CNN classifications.

Bibliography

[1] A. Abrol, M. Bhattarai, A. Fedorov, Y. Du, S. Plis, and V. Calhoun. Deep residual learning for neuroimaging: An application to predict progression to Alzheimer's disease. *Journal of Neuroscience Methods*, 339:108701, jun 2020. doi: 10.1016/j.jneumeth.2020.108701. 5.2

[2] E. Alpaydin. *Introduction to machine learning*, chapter 1 - Introduction, pages 1–20. The MIT Press, Cambridge, Massachusetts, 2014. ISBN 9780262028189. 2.1

[3] E. Alpaydin. *Introduction to machine learning*, chapter 17 - Combining Multiple Learners, pages 487–515. The MIT Press, Cambridge, Massachusetts, 2014. ISBN 9780262028189. 2.2.1, 2.2.1.1, 2.2.2

[4] E. Alpaydin. *Introduction to machine learning*, chapter 2 - Supervised Learning, pages 21–47. The MIT Press, Cambridge, Massachusetts, 2014. ISBN 9780262028189. 2.1.1.1

[5] E. Alpaydin. *Introduction to machine learning*, chapter 4 - Parametric Methods, pages 65–91. The MIT Press, Cambridge, Massachusetts, 2014. ISBN 9780262028189. 2.1.1.2

[6] E. Alpaydin. *Introduction to machine learning*, chapter 6 - Dimensionality Reduction, pages 115–160. The MIT Press, Cambridge, Massachusetts, 2014. ISBN 9780262028189. 2.3.1, 2.3.1.3, 2.3.2

[7] E. Alpaydin. *Introduction to machine learning*, chapter 9 - Decision Trees, pages 213–238. The MIT Press, Cambridge, Massachusetts, 2014. ISBN 9780262028189. (document), 2.2.2, 2.2.2, 2.2.2, 2.3, 2.2.2, 2.2.2.1

[8] Alzheimer's Disease Neuroimaging Initiative (ADNI). About ADNI. 2017. URL http://adni.loni.usc.edu/about/. [Date accessed: 11.01.2020]. 1

[9] Alzheimer's Neuroimaging Initiative (ADNI). Data Types, 2017. URL http://adni.loni.usc.edu/data-samples/data-types/. 3.1.4

[10] AlzheimerSociety. How Alzheimer's disease changes the brain. *Alzheimer Society of Canada*, 2021. [Date accessed: 04.05.2021]. (document), 2.15

[11] American Association of Neurological Surgeons. Anatomy of the Brain. 2021. URL https://www.aans.org/en/Patients/Neurosurgical-Conditions-and-Treatments/Anatomy-of-the-Brain. [Date accessed: 02.05.2021]. 2.6.1.3

[12] L. G. Apostolova, S. L. Risacher, T. Duran, E. C. Stage, N. Goukasian, J. D. West, T. M. Do, J. Grotts, H. Wilhalme, K. Nho, M. Phillips, D. Elashoff, and A. J. S. and. Associations of the Top 20 Alzheimer Disease Risk Variants With Brain Amyloidosis. *JAMA Neurology*, 75(3):328, mar 2018. doi: 10.1001/jamaneurol.2017.4198. (document), 2.7.2, 5.1, 5.1

[13] J. Ashburner, G. Barnes, C.-C. Chen, J. Daunizeau, G. Flandin, K. Friston, A. Jafarian, S. Kiebel, J. Kilner, V. Litvak, R. Moran, W. Penny, A. Razi, K. Stephan, S. Tak, P. Zeidman, D. Gitelman, R. Henson, C. Hutton, V. Glauche, J. Mattout, and C. Phillips. *SPM12 Manual*. UCL Queen Square Institute of Neurology, Jan. 2020. URL https://www.fil.ion.ucl.ac.uk/spm/. 5.2

[14] S. Basaia, F. Agosta, L. Wagner, E. Canu, G. Magnani, R. Santangelo, and M. Filippi. Automated classification of Alzheimer's disease and mild cognitive impairment using a single MRI and deep neural networks. *NeuroImage: Clinical*, 21:101645, 2019. doi: 10.1016/j.nicl.2018.101645. 1, 2.7.3, 5.2

[15] B. Beers. P-Value. *Investopedia*, 2021. URL https://www.investopedia.com/terms/p/p-value.asp. [Date accessed: 30.05.2021]. 2.8.1

[16] M. Bekkar, H. K. Djemaa, and T. A. Alitouche. Evaluation Measures for Models Assessment over Imbalanced Data Sets. *Journal of Information Engineering and Applications*, 3(10), 2013. ISSN 2225-0506. 2.8.2.1

[17] Y. Benjamini and Y. Hochberg. Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing. *Journal of the Royal Statistical Society: Series B (Methodological)*, 57(1):289–300, jan 1995. doi: 10.1111/j.2517-6161.1995.tb02031.x. 2.8.3, 2.8.3.1

[18] L. Bertram, M. McQueen, K. Mullin, D. Blacker, and R. Tanzi. The AlzGene Database. *Alzheimer Research Forum*. URL http://www.alzgene.org. [Date accessed: 15.05.2021]. 2.7.2

[19] J. G. Betts, K. A. Young, J. A. Wise, E. Johnson, B. Poe, D. H. Kruse, O. Korol, J. E. Johnson, M. Womble, and P. DeSaix. *Anatomy and Physiology* , chapter 15.3 Central Control. OpenStax, Apr. 2013. URL https://openstax.org/books/anatomy-and-physiology/pages/15-3-central-control. [Date accessed: 02.05.2021]. (document), 2.14

[20] J. H. Cole, R. P. K. Poudel, D. Tsagkrasoulis, M. W. A. Caan, C. Steves, T. D. Spector, and G. Montana. Predicting brain age with deep learning from raw imaging data results in a reliable and heritable biomarker. *NeuroImage*, 163: 115–124, dec 2017. doi: 10.1016/j.neuroimage.2017.07.059. 5.2

[21] A. Dale, B. Fischl, and M. I. Sereno. Cortical Surface-Based Analysis: I. Segmentation and Surface Reconstruction. *NeuroImage*, 9(2):179 – 194, 1999. 3.1.2

[22] A. M. Dale and M. I. Sereno. Improved Localizadon of Cortical Activity by Combining EEG and MEG with MRI Cortical Surface Reconstruction: A Linear Approach. *Journal of Cognitive Neuroscience*, 5(2):162–176, apr 1993. doi: 10.1162/jocn.1993.5.2.162. 3.1.2

[23] R. S. Desikan, F. Ségonne, B. Fischl, B. T. Quinn, B. C. Dickerson, D. Blacker, R. L. Buckner, A. M. Dale, R. P. Maguire, B. T. Hyman, M. S. Albert, and R. J. Killiany. An automated labeling system for subdividing the human cerebral cortex on MRI scans into gyral based regions of interest. *NeuroImage*, 31(3):968 – 980, 2006. ISSN 1053-8119. doi: DOI:10.1016/j.neuroimage.2006.01.021. URL http://www.sciencedirect.com/science/article/B6WNP-4JFHF4P-1/2/0ec667d4c17eafb0a7c52fa3fd5aef1c. 3.1.2

[24] R. Dinga, L. Schmaal, B. W. J. H. Penninx, D. J. Veltman, and A. F. Marquand. Controlling for effects of confounding variables on machine learning predictions. aug 2020. doi: 10.1101/2020.08.17.255034. 5.2

[25] L. Du, F. Liu, K. Liu, X. Yao, S. L. Risacher, J. Han, L. Guo, A. J. Saykin, and L. Shen. Identifying diagnosis-specific genotype–phenotype associations via joint multitask sparse canonical correlation analysis and classification. *Bioinformatics*, 36(Supplement_1):i371–i379, jul 2020. doi: 10.1093/bioinformatics/btaa434. 2.7.1

[26] J. Duchi, E. Hazan, and Y. Singer. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. *Journal of Machine Learning Research*, 12(61):2121–2159, 2011. URL http://jmlr.org/papers/v12/duchi11a.html. 2.4.5

[27] L. T. Elliott, K. Sharp, F. Alfaro-Almagro, S. Shi, K. L. Miller, G. Douaud, J. Marchini, and S. M. Smith. Genome-wide association studies of brain imaging phenotypes in UK Biobank. *Nature*, 562(7726):210–216, oct 2018. doi: 10.1038/s41586-018-0571-7. 1, 2.6.3.1

[28] B. Fischl and A. M. Dale. Measuring the thickness of the human cerebral cortex from magnetic resonance images. *Proceedings of the National Academy of Sciences of the United States of America*, 97(20):11050–11055, 2000. 3.1.2

[29] B. Fischl, M. I. Sereno, and A. Dale. Cortical Surface-Based Analysis: II: Inflation, Flattening, and a Surface-Based Coordinate System. *NeuroImage*, 9(2):195 – 207, 1999. 3.1.2

[30] B. Fischl, M. I. Sereno, R. B. Tootell, and A. M. Dale. High-resolution intersubject averaging and a coordinate system for the cortical surface. *Human Brain Mapping*, 8(4):272–284, 1999. ISSN 1097-0193. doi: 10.1002/(SICI)1097-0193(1999)8:4<272::AID-HBM10>3.0.CO;2-4. URL http://dx.doi.org/10.1002/(SICI)1097-0193(1999)8:4<272::AID-HBM10>3.0.CO;2-4. 3.1.2

[31] B. Fischl, A. Liu, and A. M. Dale. Automated manifold surgery: constructing geometrically accurate and topologically correct models of the human cerebral cortex. *IEEE Medical Imaging*, 20(1):70–80, Jan 2001. 3.1.2

[32] B. Fischl, D. H. Salat, E. Busa, M. Albert, M. Dieterich, C. Haselgrove, A. van der Kouwe, R. Killiany, D. Kennedy, S. Klaveness, A. Montillo, N. Makris, B. Rosen, and A. M. Dale. Whole brain segmentation: automated labeling of neuroanatomical structures in the human brain. *Neuron*, 33:341–355, 2002. 3.1.2

[33] B. Fischl, D. H. Salat, A. J. van der Kouwe, N. Makris, F. Ségonne, B. T. Quinn, and A. M. Dale. Sequence-independent segmentation of magnetic resonance images. *NeuroImage*, 23(Supplement 1): S69 – S84, 2004. ISSN 1053-8119. doi: DOI:10.1016/j.neuroimage.2004.07.016. URL http://www.sciencedirect.com/science/article/B6WNP-4DCMGVT-2/2/7eee26326dc63f931b826eac33becc8b. Mathematics in Brain Imaging. 3.1.2

[34] B. Fischl, A. van der Kouwe, C. Destrieux, E. Halgren, F. Ségonne, D. H. Salat, E. Busa, L. J. Seidman, J. Goldstein, D. Kennedy, V. Caviness, N. Makris, B. Rosen, and A. M. Dale. Automatically Parcellating the Human Cerebral Cortex. *Cerebral Cortex*, 14(1):11–22, 2004. doi: 10.1093/cercor/bhg087. URL http://cercor.oxfordjournals.org/content/14/1/11.abstract. 3.1.2

[35] J. B. Fraleigh. *A First Course in Abstract Algebra*. Pearson, 2014. 2.5.1

[36] FreeSurferWiki. recon-all, Dec. 2017. URL https://surfer.nmr.mgh.harvard.edu/fswiki/recon-all/. 3.1.3

[37] Y. Getachew and A. Alemu. Deep learning approach for amharic sentiment analysis. *University of Gondar*, Nov. 2018. (document), 2.4.1, 2.6

[38] M. M. González. *Atlas of Biomarkers for Alzheimer's Disease*. Springer International Publishing, July 2014. ISBN 3319079883. URL https://www.ebook.de/de/product/22426697/manuel_menendez_gonzalez_atlas_of_biomarkers_for_alzheimer_s_disease.html. 2.6.3, 2.6.3.1

[39] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*, chapter 15.2 Transfer Learning and Domain Adaption, pages 534–539. MIT Press, 2016. http://www.deeplearningbook.org. 2.4.8

[40] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*, chapter 4. Numerical Computation, pages 80–97. MIT Press, 2016. http://www.deeplearningbook.org. 2.4.1, 2.4.2

[41] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*, chapter 5.9. Stochastic Gradient Descent, pages 150–152. MIT Press, 2016. http://www.deeplearningbook.org. 2.4.2

[42] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*, chapter 6. Deep Feedforward Networks, pages 167–227. MIT Press, 2016. http://www.deeplearningbook.org. 2.4, 2.4.1, 2.4.3, 2.4.4, 2.4.4

[43] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*, chapter 7. Regularization for Deep Learning, pages 228–273. MIT Press, 2016. http://www.deeplearningbook.org. 2.2.2.1, 2.4.6

[44] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*, chapter 8. Optimization for Training Deep Models, pages 274–330. MIT Press, 2016. http://www.deeplearningbook.org. 2.4.2, 2.4.7, 2.4.7, 2.4.8

[45] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*, chapter 9. Convolutional Networks, pages 331–374. MIT Press, 2016. http://www.deeplearningbook.org. (document), 2.5, 2.5.1, 2.8, 2.5.2, 2.5.3

[46] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene Selection for Cancer Classification using Support Vector Machines. *Machine Learning*, 46(1/3):389–422, 2002. doi: 10.1023/a:1012487302797. 2.3.1.3

[47] X. Han, J. Jovicich, D. Salat, A. van der Kouwe, B. Quinn, S. Czanner, E. Busa, J. Pacheco, M. Albert, R. Killiany, P. Maguire, D. Rosas, N. Makris, A. Dale, B. Dickerson, and B. Fischl. Reliability of MRI-derived measurements of human cerebral cortical thickness: The effects of field strength, scanner upgrade and manufacturer. *NeuroImage*, 32(1):180–194, 2006. 3.1.2

[48] R. Hashemi, J. W. G. Bradley, and C. J. Lisanti. *MRI : the basics*, chapter 2 - Basic Principles of MRI, pages 16–30. Lippincott Williams & Wilkins, Philadelphia, PA, third edition, 2010. ISBN 1-60831-115-5. 2.6.2

[49] R. Hashemi, J. W. G. Bradley, and C. J. Lisanti. *MRI : the basics*, chapter 4 - T1, T2, and T2*, pages 40–43. Lippincott Williams & Wilkins, Philadelphia, PA, third edition, 2010. ISBN 1-60831-115-5. 2.6.2

[50] K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, jun 2016. doi: 10.1109/cvpr.2016.90. 2.5.4

[51] A. M. Helmenstine. Null Hypothesis Definition and Examples. *ThoughtCo.*, 2021. URL https://www.thoughtco.com/definition-of-null-hypothesis-and-examples-605436. 2.8.1

[52] E. Hosseini-Asl, G. Gimel'farb, and A. El-Baz. Alzheimer's Disease Diagnostics by a Deeply Supervised Adaptable 3D Convolutional Network. July 2016. 5.2

[53] G. Huang, Z. Liu, L. V. D. Maaten, and K. Q. Weinberger. Densely Connected Convolutional Networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, jul 2017. doi: 10.1109/cvpr.2017.243. (document), 2.5.4, 2.1, 2.5.5

[54] M. Huang, T. Nichols, C. Huang, Y. Yu, Z. Lu, R. C. Knickmeyer, Q. Feng, and H. Zhu. FVGWAS: Fast voxelwise genome wide association analysis of large-scale imaging genetic data. *NeuroImage*, 118:613–627, sep 2015. doi: 10.1016/j.neuroimage.2015.05.043. (document), 1, 2.7.3, 5.1, 5.1

[55] W. Jiang, T. Z. King, and J. A. Turner. Imaging Genetics Towards a Refined Diagnosis of Schizophrenia. *Frontiers in Psychiatry*, 10, jul 2019. doi: 10.3389/fpsyt.2019.00494. 2.7.1

[56] Z. Jin, C. Luxiang, Z. Huadong, W. Yanjiang, X. Zhiqiang, C. Hongyuan, H. Lihua, and Y. Xu. Endothelin-converting enzyme-1 promoter polymorphisms and susceptibility to sporadic late-onset Alzheimer's disease in a Chinese population. *Disease Markers*, 27:211–215, 2009. ISSN 02780240. doi: 10.3233/DMA-2009-0665. 5.1

[57] K. A. Johnson, N. C. Fox, R. A. Sperling, and W. E. Klunk. Brain Imaging in Alzheimer Disease. *Cold Spring Harbor Perspectives in Medicine*, 2 (4):a006213–a006213, jan 2012. doi: 10.1101/cshperspect.a006213. 1, 2.6.3, 2.6.3.1

[58] J. Jovicich, S. Czanner, D. Greve, E. Haley, A. van der Kouwe, R. Gollub, D. Kennedy, F. Schmitt, G. Brown, J. MacFall, B. Fischl, and A. Dale. Reliability in multi-site structural MRI studies: Effects of gradient non-linearity correction on phantom and human data. *NeuroImage*, 30(2):436 – 443, 2006. ISSN 1053-8119. doi: DOI:10.1016/j.neuroimage.2005.09.046. URL http://www.sciencedirect.com/science/article/B6WNP-4HM7S0B-2/2/4fa5ff26cad90ba3c9ed12b7e12ce3b6. 3.1.2

[59] S. Kaliyugarasan, A. Lundervold, and A. S. Lundervold. Brain Age versus Chronological Age: A Large Scale MRI and Deep Learning Investigation. 2020. doi: 10.26044/ECR2020/C-05555. 5.2

[60] S. Keel, J. Wu, P. Y. Lee, J. Scheetz, and M. He. Visualizing Deep Learning Models for the Detection of Referable Diabetic Retinopathy and Glaucoma. *JAMA Ophthalmology*, 137(3):288, mar 2019. doi: 10.1001/jamaophthalmol.2018.6035. 2.5.5

[61] T. K. Khan. *Introduction to Alzheimer's Disease Biomarkers*, pages 3–23. Elsevier, 2016. doi: 10.1016/b978-0-12-804832-0.00001-8. 2.6.3

[62] D. P. Kingma and J. Ba. Adam: A Method for Stochastic Optimization, 2017. 2.4.5, 2.4.5

[63] T. A. Knijnenburg, L. F. A. Wessels, M. J. T. Reinders, and I. Shmulevich. Fewer permutations, more accurate P-values. *Bioinformatics*, 25(12):i161–i168, may 2009. doi: 10.1093/bioinformatics/btp211. 2.8.2, 2.8.2, 5.1

[64] I. Kononenko and M. Kukar. *Machine learning and data mining : introduction to principles and algorithms*, chapter Chapter 1 - Introduction, pages 1–36. Horwood Publishing, Chichester, UK, 2007. ISBN 1904275214. 2.1, 2.4

[65] I. Kononenko and M. Kukar. *Machine learning and data mining : introduction to principles and algorithms*, chapter Chapter 3 - Machine Learning Basics, pages 59–105. Horwood Publishing, Chichester, UK, 2007. ISBN 1904275214. 2.1.1.2, 2.1.1.3, 2.1.2, 2.1.2, 2.1.2, 2.2.3

[66] I. Kononenko and M. Kukar. *Machine learning and data mining : introduction to principles and algorithms*, chapter Chapter 4 - Knowledge Representation, pages 107–130. Horwood Publishing, Chichester, UK, 2007. ISBN 1904275214. 2.2.2

[67] M. Korobov and K. Lopuhin. ELI5 Documentation Release 0.11.0. 2021. 3.2.2

[68] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, may 2017. doi: 10.1145/3065386. 2.5.4

[69] G. R. Kuperberg, M. Broome, P. K. McGuire, A. S. David, M. Eddy, F. Ozawa, D. Goff, W. C. West, S. Williams, A. van der Kouwe, D. Salat, A. Dale, and B. Fischl. Regionally localized thinning of the cerebral cortex in Schizophrenia. *Archives of General Psychiatry*, 60:878–888, 2003. 3.1.2

[70] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. doi: 10.1109/5.726791. 2.5.4

[71] W. Lin, T. Tong, Q. Gao, D. Guo, X. Du, Y. Yang, G. Guo, M. Xiao, M. Du, and X. Qu. Convolutional Neural Networks-Based MRI Image Analysis for the Alzheimer's Disease Prediction From Mild Cognitive Impairment. *Frontiers in Neuroscience*, 12, nov 2018. doi: 10.3389/fnins.2018.00777. 1, 2.7.3, 5.2

[72] J. Liu and V. D. Calhoun. A review of multivariate analyses in imaging genetics. *Frontiers in Neuroinformatics*, 8, mar 2014. doi: 10.3389/fninf.2014.00029. 1, 5.1

[73] A. S. Lundervold and A. Lundervold. An overview of deep learning in medical imaging focusing on MRI. *Zeitschrift für Medizinische Physik*, 29(2):102–127, may 2019. doi: 10.1016/j.zemedi.2018.11.002. 1, 2.4, 2.4.1, 2.4.3, 2.4.4, 2.4.6, 2.5, 2.5.2, 2.5.5, 2.7.3

[74] E. Marieb and K. Hoehn. *Human Anatomy & Physiology*, chapter 12 - The Central Nervous System, pages 450–504. Pearson, Boston, 2016. ISBN 1292096977. 2.6.1.1, 2.6.1.1, 2.6.1.2

[75] E. Masliah and D. P. Salmon. When Cognitive Decline Becomes Pathology. In *Genes, Environment and Alzheimer's Disease*, pages 29–50. Elsevier, 2016. doi: 10.1016/b978-0-12-802851-3.00002-4. 2.6.3

[76] B. H. Menze, B. M. Kelm, R. Masuch, U. Himmelreich, P. Bachert, W. Petrich, and F. A. Hamprecht. A comparison of random forest and its Gini importance with standard chemometric methods for the feature selection and classification of spectral data. *BMC Bioinformatics*, 10(1):213, 2009. doi: 10.1186/1471-2105-10-213. 1, 2.3.1.1

[77] T. Mitchell. *Machine Learning*. McGraw-Hill, New York, 1997. ISBN 0070428077. 2.1

[78] D. Mount. *Bioinformatics : sequence and genome analysis*, chapter Chapter 11. Genome Analysis, page 499. Cold Spring Harbor Laboratory Press, Cold Spring Harbor, N.Y, 2004. ISBN 0879696877. 1, 2.7.1

[79] J. C. Mueller, M. Riemenschneider, A. Schoepfer-Wendels, H. Gohlke, L. Konta, P. Friedrich, T. Illig, S. M. Laws, H. Förstl, and A. Kurz. Weak independent association signals between IDE polymorphisms, Alzheimer's disease and cognitive measures. *Neurobiology of Aging*, 28(5):727–734, may 2007. doi: 10.1016/j.neurobiolaging.2006.03.009. 5.1

[80] F. S. Nathoo, L. Kong, and H. Zhu. A review of statistical methods in imaging genetics. *Canadian Journal of Statistics*, 47(1):108–131, feb 2019. doi: 10.1002/cjs.11487. 1, 2.7.1, 2.7.3

[81] NHI - Norsk Helseinformatikk. Hjernen og nervesystemet. NHI.no, Apr. 2019. URL https://nhi.no/kroppen-var/organer/hjernen-og-nervesystemet. (norwegian) [Date accessed: 02.05.2021]. 2.6.1.1, 2.6.1.2, 2.6.1.3

[82] M. Ojala and G. C. Garriga. Permutation Tests for Studying Classifier Performance. In *2009 Ninth IEEE International Conference on Data Mining*. IEEE, dec 2009. doi: 10.1109/icdm.2009.108. 2.8.2.1

[83] T. Parr, K. Turgutlu, C. Csiszar, and J. Howard. Beware Default Random Forest Importances. Mar. 2018. URL https://explained.ai/rf-importance/index.html. [Date accessed: 07.05.2021]. 2.3.1.1, 2.3.1.2

[84] P. Parviainen. Lecture 2. Basic concepts and k-Nearest Neighbours. In *Introduction to Machine Learning*. University of Bergen, Aug. 2019. (document), 2.1

[85] P. Parviainen. Lectures 9-10. neural networks and deep learning. In *Introduction to Machine Learning*. University of Bergen, Oct. 2019. (document), 2.4, 2.4, 2.5, 2.7, 2.9, 2.10

[86] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in PyTorch. 2017. 3.2.4

[87] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011. 3.2.1

[88] M. Peters. Alzheimer's Disease Big Data DREAM Challenge 1, 2014. [Date accessed: 12.03.2021]. 3.1.1, 3.1.2

[89] B. T. Polyak and A. B. Juditsky. Acceleration of Stochastic Approximation by Averaging. *SIAM Journal on Control and Optimization*, 30(4):838–855, jul 1992. doi: 10.1137/0330046. 2.4.5

[90] M. Pominova, E. Kondrateva, M. Sharaev, A. Bernstein, S. Pavlov, and E. Burnaev. 3D Deformable Convolutions for MRI Classification. In *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*. IEEE, dec 2019. doi: 10.1109/icmla.2019.00278. 1, 2.5.5, 3.1.2, 5.2

[91] N. Ponomareva, T. Andreeva, V. Fokin, S. Illarioshkin, and E. Rogaev. Linking EEGs, Alzheimer disease, and the phosphatidylinositol-binding clathrin assembly protein (PICALM) gene. In *Genetics, Neurology, Behavior, and Diet in Dementia*, pages 41–55. Elsevier, 2020. doi: 10.1016/b978-0-12-815868-5.00003-7. 2.7.2

[92] D. F. V. Pîrşcoveanu, I. Pirici, V. Tudorică, T. A. Bălşeanu, V. C. Albu, S. Bondari, A. M. Bumbea, and M. Pîrşcoveanu. Tau protein in neurodegenerative diseases - a review. *Romanian journal of morphology and embryology = Revue roumaine de morphologie et embryologie*, 58:1141–1150, 2017. ISSN 2066-8279. 2.6.3

[93] Project MONAI. MONAI Medical Open Network for AI. 2021. URL https://monai.io. 3.2.5

[94] S. Pujara. Frontal Lobe and Pathology. *Neurosurgery for Medical Students*. URL https://myneurosurg.com/cranial-anatomy/frontal-lobe-and-pathology/. [Date accessed: 02.05.2021]. (document), 2.13

[95] M. Reuter, H. D. Rosas, and B. Fischl. Highly Accurate Inverse Consistent Registration: A Robust Approach. *NeuroImage*, 53(4):1181–1196, 2010. doi: 10.1016/j.neuroimage.2010.07.020. URL http://dx.doi.org/10.1016/j.neuroimage.2010.07.020. 3.1.2

[96] M. Reuter, N. J. Schmansky, H. D. Rosas, and B. Fischl. Within-Subject Template Estimation for Unbiased Longitudinal Image Analysis. *NeuroImage*, 61(4):1402–1418, 2012. doi: 10.1016/j.neuroimage.2012.02.084. URL http://dx.doi.org/10.1016/j.neuroimage.2012.02.084. 3.1.2

[97] H. D. Rosas, A. K. Liu, S. Hersch, M. Glessner, R. J. Ferrante, D. H. Salat, A. van der Kouwe, B. G. Jenkins, A. M. Dale, and B. Fischl. Regional and progressive thinning of the cortical ribbon in Huntington's disease. *Neurology*, 58(5):695–701, 2002. URL http://www.neurology.org/content/58/5/695.abstract. 3.1.2

[98] J. Ruiz, M. Mahmud, M. Modasshir, M. S. Kaiser, and for the Alzheimer's Disease Neuroimaging In. 3D DenseNet Ensemble in 4-Way Classification of Alzheimer's Disease. In *Brain Informatics*, pages 85–96. Springer International Publishing, 2020. doi: 10.1007/978-3-030-59277-6_8. (document), 2.7.3, 3.4, 5.2, 5.3

[99] D. Salat, R. Buckner, A. Snyder, D. N. Greve, R. Desikan, E. Busa, J. Morris, A. Dale, and B. Fischl. Thinning of the cerebral cortex in aging. *Cerebral Cortex*, 14:721–730, 2004. 3.1.2

[100] S. Sarraf and G. Tofighi. Classification of Alzheimer's Disease Structural MRI Data by Deep Learning Convolutional Neural Networks. July 2016. 5.2

[101] S. Seabold and J. Perktold. statsmodels: Econometric and statistical modeling with python. In *9th Python in Science Conference*, 2010. 3.2.3

[102] F. Segonne, A. M. Dale, E. Busa, M. Glessner, D. Salat, H. K. Hahn, and B. Fischl. A hybrid approach to the skull stripping problem in MRI. *NeuroImage*, 22(3):1060 – 1075, 2004. ISSN 1053-8119. doi: DOI:10.1016/j.neuroimage.2004.03.032. URL http://www.sciencedirect.com/science/article/B6WNP-4CF5CNY-1/2/33cc73136f06f019b2c11023e7a95341. 3.1.2

[103] F. Segonne, J. Pacheco, and B. Fischl. Geometrically accurate topology-correction of cortical surfaces using nonseparating loops. *IEEE Trans Med Imaging*, 26:518–529, 2007. 3.1.2

[104] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization. In *2017 IEEE International Conference on Computer Vision (ICCV)*. IEEE, oct 2017. doi: 10.1109/iccv.2017.74. 2.5.5, 5.2

[105] H. Shahamat and M. S. Abadeh. Brain MRI analysis using a deep learning based evolutionary approach. *Neural Networks*, 126:218–234, jun 2020. doi: 10.1016/j.neunet.2020.03.017. 1, 2.7.3

[106] L. Shen, S. Kim, S. L. Risacher, K. Nho, S. Swaminathan, J. D. West, T. Foroud, N. Pankratz, J. H. Moore, C. D. Sloan, M. J. Huentelman, D. W. Craig, B. M. DeChairo, S. G. Potkin, C. R. Jack, M. W. Weiner, and A. J. Saykin. Whole genome association study of brain-wide imaging phenotypes for identifying quantitative trait loci in MCI and AD: A study of the ADNI cohort. *NeuroImage*, 53(3):1051–1063, nov 2010. doi: 10.1016/j.neuroimage.2010.01.042. 2.7.2

[107] J. M. Shulman, K. Chen, B. T. Keenan, L. B. Chibnik, A. Fleisher, P. Thiyyagura, A. Roontiva, C. McCabe, N. A. Patsopoulos, J. J. Corneveaux, L. Yu, M. J. Huentelman, D. A. Evans, J. A. Schneider, E. M. Reiman, P. L. D. Jager, and D. A. Bennett. Genetic Susceptibility for Alzheimer Disease Neuritic Plaque Pathology. *JAMA Neurology*, 70(9):1150, sep 2013. doi: 10.1001/jamaneurol.2013.2815. 2.7.2

[108] G. Sienski, P. Narayan, J. M. Bonner, N. Kory, S. Boland, A. A. Arczewska, W. T. Ralvenius, L. Akay, E. Lockshin, L. He, B. Milo, A. Graziosi, V. Baru, C. A. Lewis, M. Kellis, D. M. Sabatini, L.-H. Tsai, and S. Lindquist. APOE4 disrupts intracellular lipid homeostasis in human iPSC-derived glia. *Science Translational Medicine*, 13(583):eaaz4564, Mar. 2021. doi: 10.1126/scitranslmed.aaz4564. 3.1.4

[109] J. Sled, A. Zijdenbos, and A. Evans. A nonparametric method for automatic correction of intensity nonuniformity in MRI data. *IEEE Trans Med Imaging*, 17:87–97, 1998. 3.1.2

[110] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller. Striving for Simplicity: The All Convolutional Net. Dec. 2014. 5.2

[111] J. L. Stein, X. Hua, S. Lee, A. J. Ho, A. D. Leow, A. W. Toga, A. J. Saykin, L. Shen, T. Foroud, N. Pankratz, M. J. Huentelman, D. W. Craig, J. D. Gerber, A. N. Allen, J. J. Corneveaux, B. M. DeChairo, S. G. Potkin, M. W. Weiner, and P. M. Thompson. Voxelwise genome-wide association study (vGWAS). *NeuroImage*, 53(3):1160–1174, nov 2010. doi: 10.1016/j.neuroimage.2010.02.032. 2.7.3

[112] J. D. Storey and R. Tibshirani. Statistical significance for genomewide studies. *Proceedings of the National Academy of Sciences*, 100(16):9440–9445, jul 2003. doi: 10.1073/pnas.1530509100. 2.8.3

[113] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, jun 2015. doi: 10.1109/cvpr.2015.7298594. 2.5.4

[114] B. Szymik. What's Your Brain Doing? ASU - Ask A Biologist, May 2011. URL https://askabiologist.asu.edu/brain-regions. [Date accessed: 02.05.2021]. (document), 2.12

[115] H. Tang, E. Yao, G. Tan, and X. Guo. A Fast and Accurate 3D Fine-Tuning Convolutional Neural Network for Alzheimer's Disease Diagnosis. In *Communications in Computer and Information Science*, pages 115–126. Springer Singapore, 2018. doi: 10.1007/978-981-13-2122-1_9. 2.7.3, 5.2

[116] T. Tieleman and G. Hinton. Lecture 6.5 - RMSProp. In *Neural Networks for Machine Learning*. 2012. Technical report. 2.4.5

[117] R. M. Twyman. Single-Nucleotide Polymorphism (SNP) Analysis. In *Encyclopedia of Neuroscience*, pages 871–875. Elsevier, 2009. doi: 10.1016/b978-008045046-9.00866-4. 2.7.1

[118] R. C. Venkata and D. Ghersi. Biological Pathway Analysis. In *Encyclopedia of Bioinformatics and Computational Biology*, pages 1067–1070. Elsevier, 2019. doi: 10.1016/b978-0-12-809633-8.20476-7. 1, 2.7.1

[119] H. Wang, F. Nie, H. Huang, S. Kim, K. Nho, S. L. Risacher, A. J. Saykin, and L. Shen. Identifying quantitative trait loci via group-sparse multitask regression and feature selection: an imaging genetics study of the ADNI cohort. *Bioinformatics*, 28(2):229–237, dec 2012. doi: 10.1093/bioinformatics/btr649. 1, 2.7.3

[120] H. Wang, F. Nie, H. Huang, J. Yan, S. Kim, K. Nho, S. L. Risacher, A. J. Saykin, and L. Shen. From phenotype to genotype: an association study of longitudinal phenotypic markers to Alzheimer's disease relevant SNPs. *Bioinformatics*, 28 (18):i619–i625, sep 2012. doi: 10.1093/bioinformatics/bts411. 2.7.3

[121] H. Wang, Y. Shen, S. Wang, T. Xiao, L. Deng, X. Wang, and X. Zhao. Ensemble of 3D densely connected convolutional network for diagnosis of mild cognitive impairment and Alzheimer's disease. *Neurocomputing*, 333:145–156, mar 2018. doi: 10.1016/j.neucom.2018.12.018. 2.7.3, 5.2

[122] H. Wang, Z. Wu, and E. P. Xing. Removing Confounding Factors Associated Weights in Deep Neural Networks Improves the Prediction Accuracy for Healthcare Applications. *Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing*, 24:54–65, 2019. ISSN 2335-6936. 5.2

[123] W. Xu, J. Xu, Y. Wang, H. Tang, Y. Deng, R. Ren, G. Wang, W. Niu, J. Ma, Y. Wu, J. Zheng, S. Chen, and J. Ding. The genetic variation of SORCS1 is associated with late-onset alzheimer's disease in chinese han population. *PLoS ONE*, 8(5):e63621, may 2013. doi: 10.1371/journal.pone.0063621. 5.1

[124] A. D. Yates, P. Achuthan, W. Akanni, J. Allen, J. Allen, J. Alvarez-Jarreta, M. R. Amode, I. M. Armean, A. G. Azov, R. Bennett, J. Bhai, K. Billis, S. Boddu, J. C. Marugán, C. Cummins, C. Davidson, K. Dodiya, R. Fatima, A. Gall, C. G. Giron, L. Gil, T. Grego, L. Haggerty, E. Haskell, T. Hourlier, O. G. Izuogu, S. H. Janacek, T. Juettemann, M. Kay, I. Lavidas, T. Le, D. Lemos, J. G. Martinez, T. Maurel, M. McDowall, A. McMahon, S. Mohanan, B. Moore, M. Nuhn, D. N. Oheh, A. Parker, A. Parton, M. Patricio, M. P. Sakthivel, A. I. A. Salam, B. M. Schmitt, H. Schuilenburg, D. Sheppard, M. Sycheva, M. Szuba, K. Taylor, A. Thormann, G. Threadgold, A. Vullo, B. Walts, A. Winterbottom, A. Zadissa, M. Chakiachvili, B. Flint, A. Frankish, S. E. Hunt, G. IIsley, M. Kostadima, N. Langridge, J. E. Loveland, F. J. Martin, J. Morales, J. M. Mudge, M. Muffato, E. Perry, M. Ruffier, S. J. Trevanion, F. Cunningham, K. L. Howe, D. R. Zerbino, and P. Flicek. Ensembl 2020. *Nucleic Acids Research*, nov 2019. doi: 10.1093/nar/gkz966. 3.3.6

[125] M. D. Zeiler and R. Fergus. Visualizing and Understanding Convolutional Networks. In *Computer Vision – ECCV 2014*, pages 818–833. Springer International Publishing, 2014. doi: 10.1007/978-3-319-10590-1_53. 2.5.5, 5.2

[126] Q. Zhao, E. Adeli, and K. M. Pohl. Training confounder-free deep learning models for medical applications. *Nature Communications*, 11(1), nov 2020. doi: 10.1038/s41467-020-19784-9. 5.2

Appendix 1 – Tables for Methods

Table A.1.1: Classes and methods from scikit-learn used in the experiments. For documentation, see https://scikit-learn.org.

Type	Name	Description	Default parameters
Class	RandomForest-Classifier	Creates a random forest classifier instance.	n_estimators=100, criterion='gini', max_depth=None, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features='auto', max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, bootstrap=True, oob_score=False, n_jobs=None, random_state=None, verbose=0, warm_start=False, class_weight=None, ccp_alpha=0.0, max_samples=None
Class	DummyClassifier	Creates a dummy classifier that makes predictions using simple rules. Can be used as a baseline comparison, but not fit for real problems.	strategy='prior', random_state=None, constant=None
Class	RFECV	Creates feature ranking instance with recursive feature elimination and cross-validated selection of the best number of features.	step=1, min_features_to_select=1, cv=None, scoring=None, verbose=0, n_jobs=None, importance_getter='auto'
Class	RFE	Creates feature ranking instance with recursive feature elimination.	n_features_to_select=None, step=1, verbose=0, importance_getter='auto'
Class	StratifiedKFold	Creates stratified K-Folds cross-validator instance. Provides train/test indices to split data in train/test sets.	n_splits=5, shuffle=False, random_state=None
Class	MinMaxScaler	Transform features by scaling each feature to a given range.	feature_range=0, 1, copy=True, clip=False
Method	accuracy_score	Computes accuracy classification score of input.	normalize=True, sample_weight=None

Method	f1_score	Computes the F1 score of the input.	labels=None, pos_label=1, average='binary', sample_weight=None, zero_division='warn'
Method	permutation_test_score	Performes permutation testing for the given model and data. Returns the score, permutation scores and pvalue.	groups=None, cv=None, n_permutations=100, n_jobs=None, random_state=0, verbose=0, scoring=None, fit_params=None
Method	confusion_matrix	Computes the confusion matrix to evaluate the accuracy of a classification.	labels=None, sample_weight=None, normalize=None
Method	precision_score	Computes the precision of a classification.	labels=None, pos_label=1, average='binary', sample_weight=None, zero_division='warn'
Method	recall_score	Computes the recall a classifiaction.	labels=None, pos_label=1, average='binary', sample_weight=None, zero_division='warn'
Class	ConfusionMatrix-Display	Confusion Matrix visualization.	display_labels=None
Method	cross_val_score	Evaluates a score by cross-validation.	groups=None, scoring=None, cv=None, n_jobs=None, verbose=0, fit_params=None, pre_dispatch='2*n_jobs', error_score=nan
Method	train_test_split	Split arrays or matrices into random train and test subsets.	test_size=None, train_size=None, random_state=None, shuffle=True, stratify=None

Table A.1.2: Classes and methods from MONAI used in the experiments. For documentation, see https://docs.monai.io.

Type	Name	Description	Default parameters
Class	densenet121	Creates Densenet121 model instance	init_features=64, growth_rate=32, block_config=(6, 12, 24, 16), bn_size=4, dropout_prob=0.0, pretrained = False, progress = True
Class	DataLoader	Generates images/labels from dataset	num_workers=0
Class	ArrayDataset	Creates dataset for segmentation and classification tasks based on input data and transforms	img_transform=None, seg=None, seg_transform=None, labels=None, label_transform=None

Class	Compose	Provides the ability to chain a series of transformation calls in a sequence	transforms=None
Class	AddChannel	Adds a 1-length channel dimension to the input image	
Class	Resize	Scaling the input image to given spatial size.	mode= <InterpolateMode.AREA: 'area'>, align_corners=None
Class	ScaleIntensity	Scaling the intensity of the input image.	minv=0.0, maxv=1.0, factor=None
Class	ToTensor	Converts input image to a tensor	
Class	LoadImage	Load image file or files from provided path based on reader. Chooses reader automatically from input type.	reader=None, image_only=False, dtype=<class 'numpy.float32'>
Class	RandZoom	Randomly zooms input arrays with given probability within given zoom range.	prob=0.1, min_zoom=0.9, max_zoom=1.1, mode= <InterpolateMode.AREA: 'area'>, padding_mode= <NumpyPadMode.EDGE: 'edge'>, align_corners=None, keep_size=True
Class	Occlusion-Sensitivity	Computes the occlusion sensitivity for a model's prediction of a given image	pad_val=None, mask_size=15, n_batch=128, stride=1, upsampler= functools.partial(<function default_upsampler>, align_corners=True), verbose=True
Method	set_determin-ism	Sets random seed for modules to enable deterministic training.	seed=4294967295, additional_settings=None

Table A.1.3: Chromosome and positional data for genes used in SNP data selection in Phase 1, GWAS 2 described in section 3.3.6. Human reference genome GRCh37.

Gene name	Chromosome	GRCh37 start pos (bp)	GRCh37 end pos (bp)
PRNP	20	4666996	4682236
CST3	20	23607343	23619110
IL33	9	6215786	6257983
CLU	8	27454434	27472217
OTC	23	38211839	38280699
DAPK1	9	90112143	90323549
NEDD9	6	11183531	11382581

PGBD1	6	28249349	28270326
TFAM	10	60145105	60158980
ADAM10	15	58881008	59041990
TNF	6	31543342	31546113
PCDH11X	23	91034260	91878229
LDLR	19	11200138	11244496
CH25H	10	90965386	90967074
IDE	10	94211441	94333853
MTHFR	1	11845780	11866512
ENTPD7	10	101419266	101470998
CALHM1	10	105212997	105218657
SORCS1	10	108333421	108924292
GAPDHS	19	36024357	36036221
ECE1	1	21543740	21672065
CCR2	3	46395235	46402431
EXOC3L2	19	45715628	45748665
IL1A	2	113531502	113542070
IL1B	2	113587328	113594393
BIN1	2	127805603	127864864
TF	3	133464884	133515485
GAB2	11	77926339	78129394
PICALM	11	85668218	85780924
SORL1	11	121323023	121504472
CHRNB2	1	154540254	154552489
CR1	1	207669492	207815110
TNK1	17	7283853	7293093
THRA	17	38214543	38250120
GRN	17	42422614	42430474
ACE	17	61554422	61575741

Table A.1.4: The set of features used for feature selection 2 of phase 1, GWAS 2, described in section 3.3.6.2, and the set of their correlated features.

Nr.	Feature	Correlated features
1	Volume.ctx-lh-unknown	
2	Volume.ctx-lh-bankssts	
3	Volume.ctx-lh-caudalanteriorcingulate	
4	Volume.ctx-lh-caudalmiddlefrontal	Volume.wm-lh-caudalmiddlefrontal, Volume.ctx-rh-caudalmiddlefrontal
5	Volume.ctx-lh-cuneus	Volume.ctx-lh-pericalcarine, Volume.ctx-rh-cuneus
6	Volume.ctx-lh-entorhinal	
7	Volume.ctx-lh-fusiform	Volume.ctx-rh-fusiform

8	Volume.ctx-lh-inferiorparietal	Volume.ctx-lh-middletemporal, Volume.ctx-lh-precuneus, Volume.wm-lh-inferiorparietal, Volume.ctx-rh-inferiorparietal
9	Volume.ctx-lh-inferiortemporal	Volume.ctx-rh-inferiortemporal
10	Volume.ctx-lh-isthmuscingulate	Volume.wm-lh-isthmuscingulate
11	Volume.ctx-lh-lateraloccipital	Volume.ctx-rh-lateraloccipital
12	Volume.ctx-lh-lateralorbitofrontal	Volume.ctx-lh-parsorbitalis, Volume.ctx-rh-lateralorbitofrontal, Volume.ctx-rh-parsorbitalis
13	Volume.ctx-lh-lingual	Volume.wm-lh-lingual, Volume.ctx-rh-lingual
14	Volume.ctx-lh-medialorbitofrontal	Volume.ctx-rh-medialorbitofrontal
15	Volume.ctx-lh-parahippocampal	Volume.wm-lh-parahippocampal, Volume.ctx-rh-parahippocampal
16	Volume.ctx-lh-paracentral	
17	Volume.ctx-lh-parsopercularis	Volume.wm-lh-parsopercularis
18	Volume.ctx-lh-parstriangularis	Volume.wm-lh-parstriangularis
19	Volume.ctx-lh-postcentral	Volume.ctx-lh-precentral, Volume.ctx-rh-postcentral
20	Volume.ctx-lh-posteriorcingulate	Volume.wm-lh-posteriorcingulate
21	Volume.ctx-lh-rostralanteriorcingulate	
22	Volume.ctx-lh-rostralmiddlefrontal	Volume.ctx-lh-precuneus, Volume.ctx-lh-superiorfrontal, Volume.ctx-rh-rostralmiddlefrontal, Volume.ctx-rh-superiorfrontal
23	Volume.ctx-lh-superiorparietal	Volume.ctx-lh-precuneus, Volume.ctx-rh-precuneus, Volume.ctx-rh-superiorparietal
24	Volume.ctx-lh-superiortemporal	Volume.ctx-rh-superiortemporal
25	Volume.ctx-lh-supramarginal	Volume.wm-lh-supramarginal, Volume.ctx-rh-supramarginal
26	Volume.ctx-lh-frontalpole	
27	Volume.ctx-lh-temporalpole	
28	Volume.ctx-lh-transversetemporal	
29	Volume.ctx-lh-insula	Volume.wm-lh-insula, Volume.ctx-rh-insula
30	Volume.Left-Cerebral-White-Matter	Volume.Right-Cerebral-White-Matter
31	Volume.Left-Lateral-Ventricle	Volume.Left-Inf-Lat-Vent, Volume.Right-Lateral-Ventricle, Volume.CSF
32	Volume.Left-Thalamus-Proper	Volume.Right-Thalamus-Proper, Volume.Brain-Stem

33	Volume.Left-Caudate	Volume.Right-Caudate
34	Volume.Left-Putamen	Volume.Right-Putamen
35	Volume.Left-Pallidum	
36	Volume.Left-Hippocampus	Volume.Left-Amygdala, Volume.Right-Hippocampus
37	Volume.Left-Accumbens-area	Volume.Right-Accumbens-area
38	Volume.Left-VentralDC	Volume.Right-VentralDC, Volume.Brain-Stem
39	Volume.Left-vessel	
40	Volume.Left-choroid-plexus	Volume.Right-choroid-plexus
41	Volume.wm-lh-bankssts	
42	Volume.wm-lh-caudalanteriorcingulate	
43	Volume.wm-lh-cuneus	Volume.wm-rh-cuneus
44	Volume.wm-lh-entorhinal	
45	Volume.wm-lh-fusiform	Volume.wm-rh-fusiform
46	Volume.wm-lh-inferiortemporal	Volume.wm-lh-middletemporal, Volume.wm-lh-superiorfrontal, Volume.wm-rh-inferiortemporal, Volume.wm-rh-middletemporal
47	Volume.wm-lh-lateraloccipital	Volume.wm-rh-lateraloccipital
48	Volume.wm-lh-lateralorbitofrontal	Volume.wm-lh-rostralmiddlefrontal, Volume.wm-lh-superiorfrontal, Volume.wm-rh-lateralorbitofrontal, Volume.wm-rh-rostralmiddlefrontal, Volume.wm-rh-superiorfrontal, Volume.wm-rh-insula
49	Volume.wm-lh-medialorbitofrontal	
50	Volume.wm-lh-paracentral	Volume.wm-lh-precentral, Volume.wm-rh-paracentral, Volume.wm-rh-precentral, Volume.wm-rh-superiorfrontal
51	Volume.wm-lh-parsorbitalis	
52	Volume.wm-lh-pericalcarine	Volume.ctx-lh-pericalcarine, Volume.wm-lh-lingual, Volume.wm-rh-lingual, Volume.wm-rh-pericalcarine
53	Volume.wm-lh-postcentral	Volume.wm-lh-precentral, Volume.wm-lh-superiorfrontal, Volume.wm-rh-postcentral, Volume.wm-rh-precentral, Volume.wm-rh-superiorfrontal
54	Volume.wm-lh-precuneus	Volume.wm-lh-rostralmiddlefrontal, Volume.wm-lh-superiorfrontal, Volume.wm-rh-precuneus, Volume.wm-rh-superiorfrontal, Volume.wm-rh-superiorparietal
55	Volume.wm-lh-rostralanteriorcingulate	
56	Volume.wm-lh-superiorparietal	Volume.wm-rh-superiorparietal
57	Volume.wm-lh-superiortemporal	Volume.wm-lh-superiorfrontal, Volume.wm-lh-supramarginal, Volume.wm-rh-superiorfrontal, Volume.wm-rh-superiortemporal
58	Volume.wm-lh-frontalpole	

59	Volume.wm-lh-temporalpole	Volume.wm-rh-temporalpole
60	Volume.wm-lh-transversetemporal	
61	Volume.Left-UnsegmentedWhiteMatter	Volume.wm-lh-superiorfrontal, Volume.wm-rh-superiorfrontal, Volume.Right-UnsegmentedWhiteMatter, Volume.Brain-Stem
62	Volume.ctx-rh-unknown	
63	Volume.ctx-rh-bankssts	Volume.wm-rh-bankssts
64	Volume.ctx-rh-caudalanteriorcingulate	Volume.wm-rh-caudalanteriorcingulate
65	Volume.ctx-rh-entorhinal	
66	Volume.ctx-rh-isthmuscingulate	Volume.wm-rh-isthmuscingulate
67	Volume.ctx-rh-middletemporal	Volume.ctx-lh-middletemporal, Volume.ctx-rh-inferiorparietal, Volume.ctx-rh-inferiortemporal
68	Volume.ctx-rh-paracentral	Volume.wm-rh-paracentral
69	Volume.ctx-rh-parsopercularis	Volume.wm-rh-parsopercularis
70	Volume.ctx-rh-parstriangularis	Volume.wm-rh-parstriangularis
71	Volume.ctx-rh-pericalcarine	Volume.ctx-lh-pericalcarine, Volume.wm-rh-pericalcarine
72	Volume.ctx-rh-posteriorcingulate	Volume.wm-rh-posteriorcingulate
73	Volume.ctx-rh-precentral	Volume.ctx-lh-precentral, Volume.ctx-lh-superiorfrontal, Volume.ctx-rh-superiorfrontal
74	Volume.ctx-rh-rostralanteriorcingulate	
75	Volume.ctx-rh-frontalpole	
76	Volume.ctx-rh-temporalpole	
77	Volume.ctx-rh-transversetemporal	
78	Volume.Right-Inf-Lat-Vent	Volume.Left-Inf-Lat-Vent, Volume.Right-Lateral-Ventricle
79	Volume.Right-Pallidum	
80	Volume.Right-Amygdala	Volume.Left-Amygdala, Volume.Right-Hippocampus
81	Volume.Right-vessel	
82	Volume.wm-rh-caudalmiddlefrontal	Volume.wm-lh-caudalmiddlefrontal, Volume.ctx-rh-caudalmiddlefrontal
83	Volume.wm-rh-entorhinal	
84	Volume.wm-rh-inferiorparietal	Volume.wm-lh-inferiorparietal

85	Volume.wm-rh-medialorbitofrontal	Volume.wm-lh-superiorfrontal, Volume.wm-rh-superiorfrontal
86	Volume.wm-rh-parahippocampal	Volume.wm-lh-parahippocampal
87	Volume.wm-rh-parsorbitalis	
88	Volume.wm-rh-rostralanteriorcingulate	
89	Volume.wm-rh-supramarginal	Volume.wm-lh-supramarginal, Volume.ctx-rh-supramarginal, Volume.wm-rh-superiortemporal
90	Volume.wm-rh-frontalpole	
91	Volume.wm-rh-transversetemporal	
92	Volume.Optic-Chiasm	

Appendix 2 – Occlusion Sensitivity Results

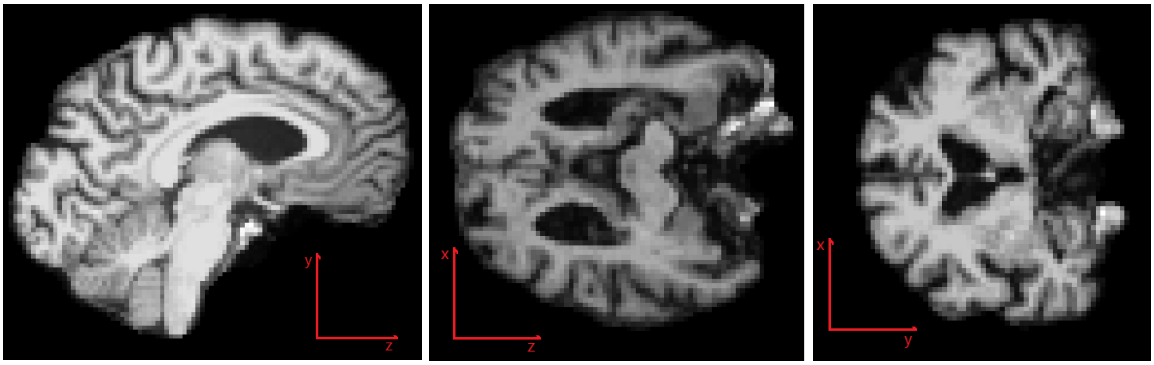


Figure A.2.1: Image axes used for the occlusion plots. Left: image is sliced along the x axis. Middle: image is sliced along the y axis. Right: image is sliced along the z axis.

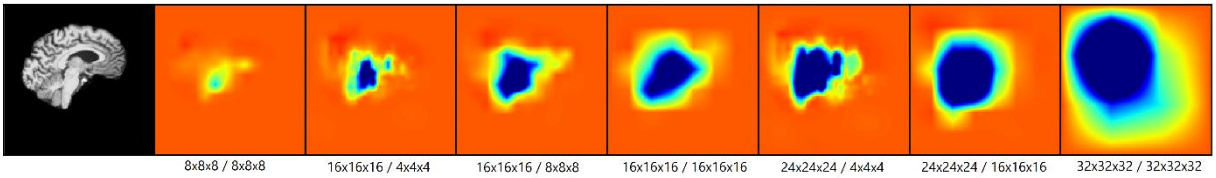


Figure A.2.2: Effect of mask size and stride size on occlusion sensitivity.

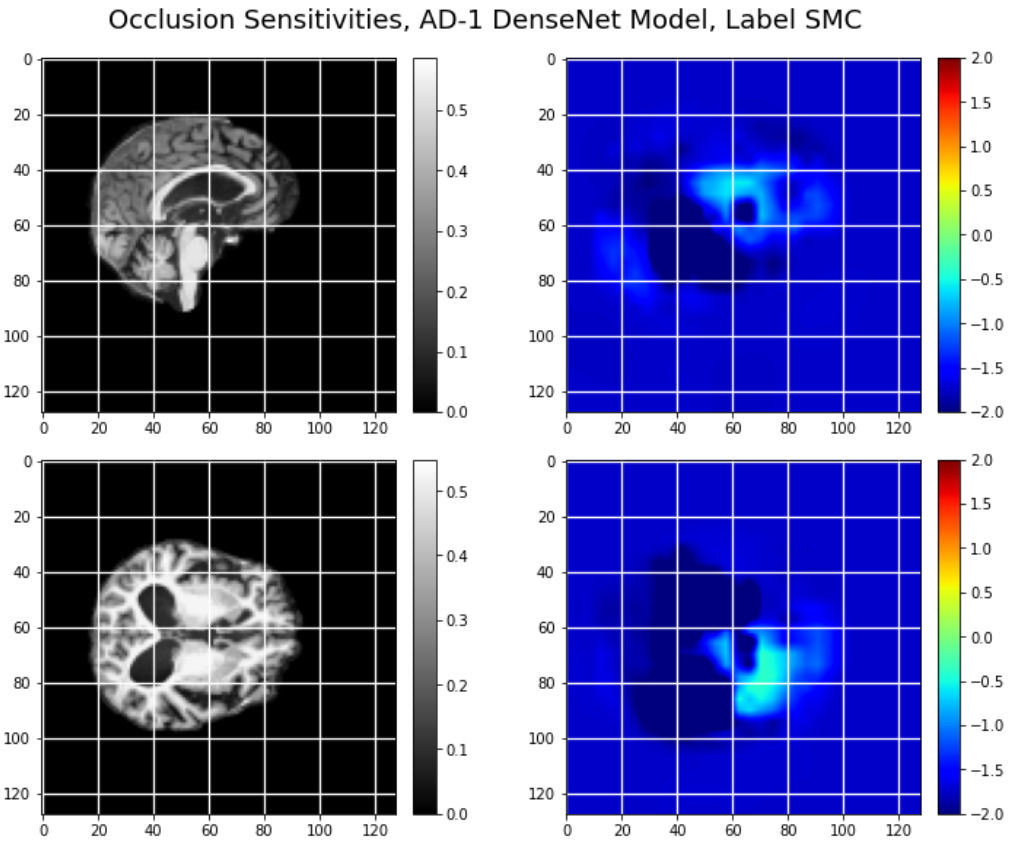


Figure A.2.3: Occlusion sensitivity maps for AD-1 DenseNet model (5-way classification of AD, no pretraining), for prediction of label "SMC". To the left is the image slice, to the right is the occlusion map of this image slice. Slices have been made along the following axis (from top): x=60 and y=50.

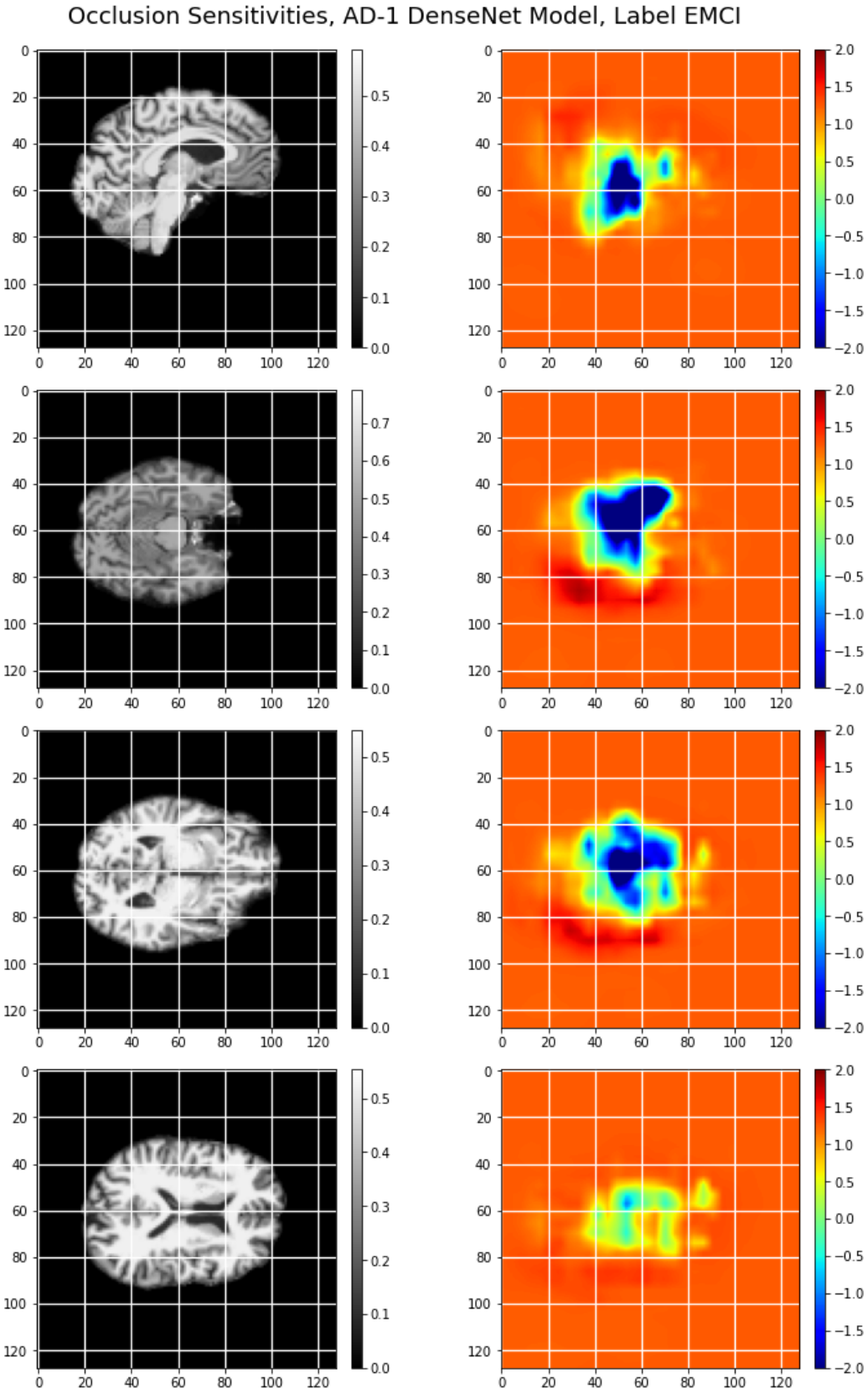


Figure A.2.4: Occlusion sensitivity maps for AD-1 DenseNet model (5-way classification of AD, no pretraining), for prediction of label "EMCI". To the left is the image slice, to the right is the occlusion map of this image slice. Slices have been made along the following axis (from top): x=60, y=60, y=50 and y=40.

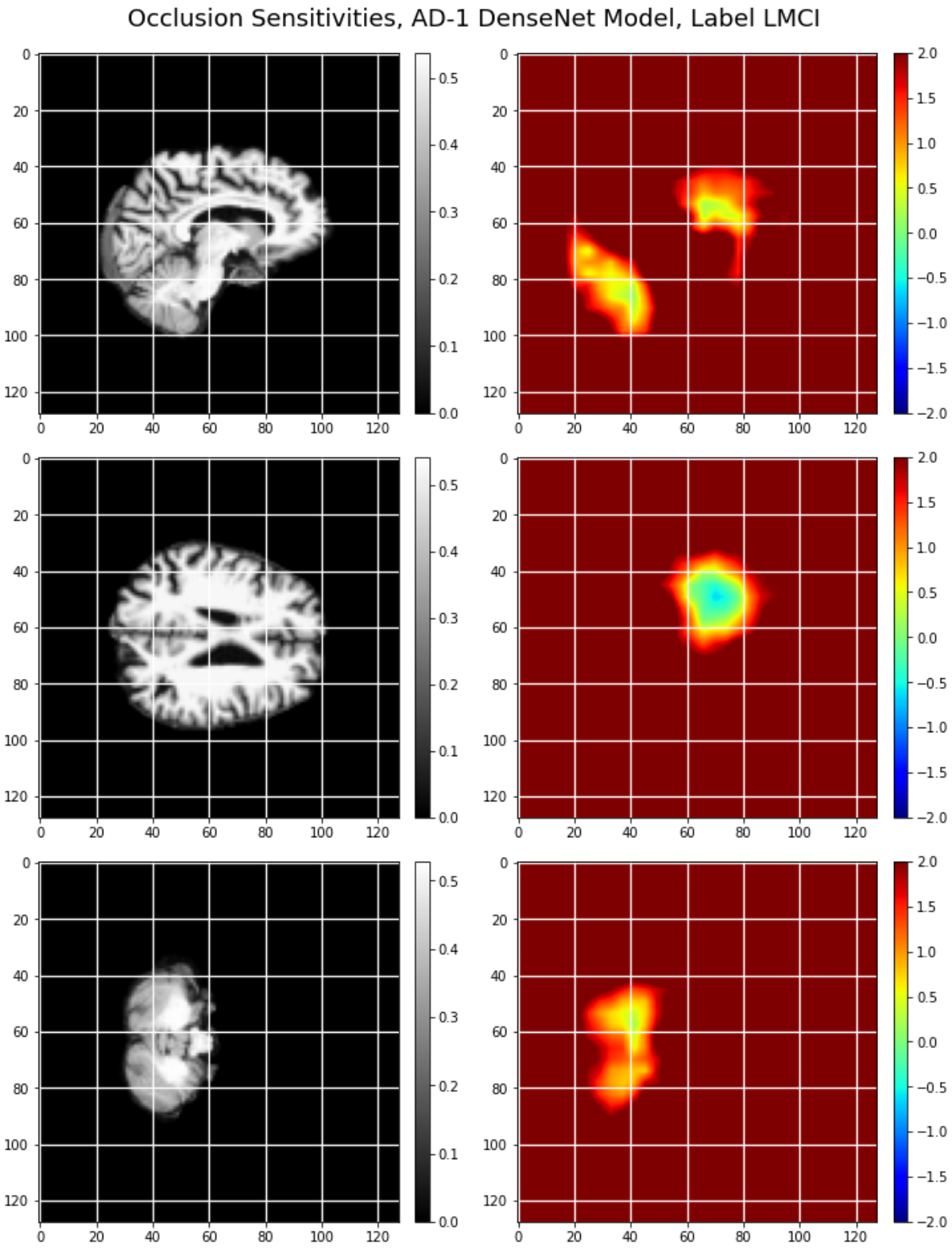


Figure A.2.5: Occlusion sensitivity maps for AD-1 DenseNet model (5-way classification of AD, no pretraining), for prediction of label "LMCI". To the left is the image slice, to the right is the occlusion map of this image slice. Slices have been made along the following axis (from top): x=55, y=50 and y=85.

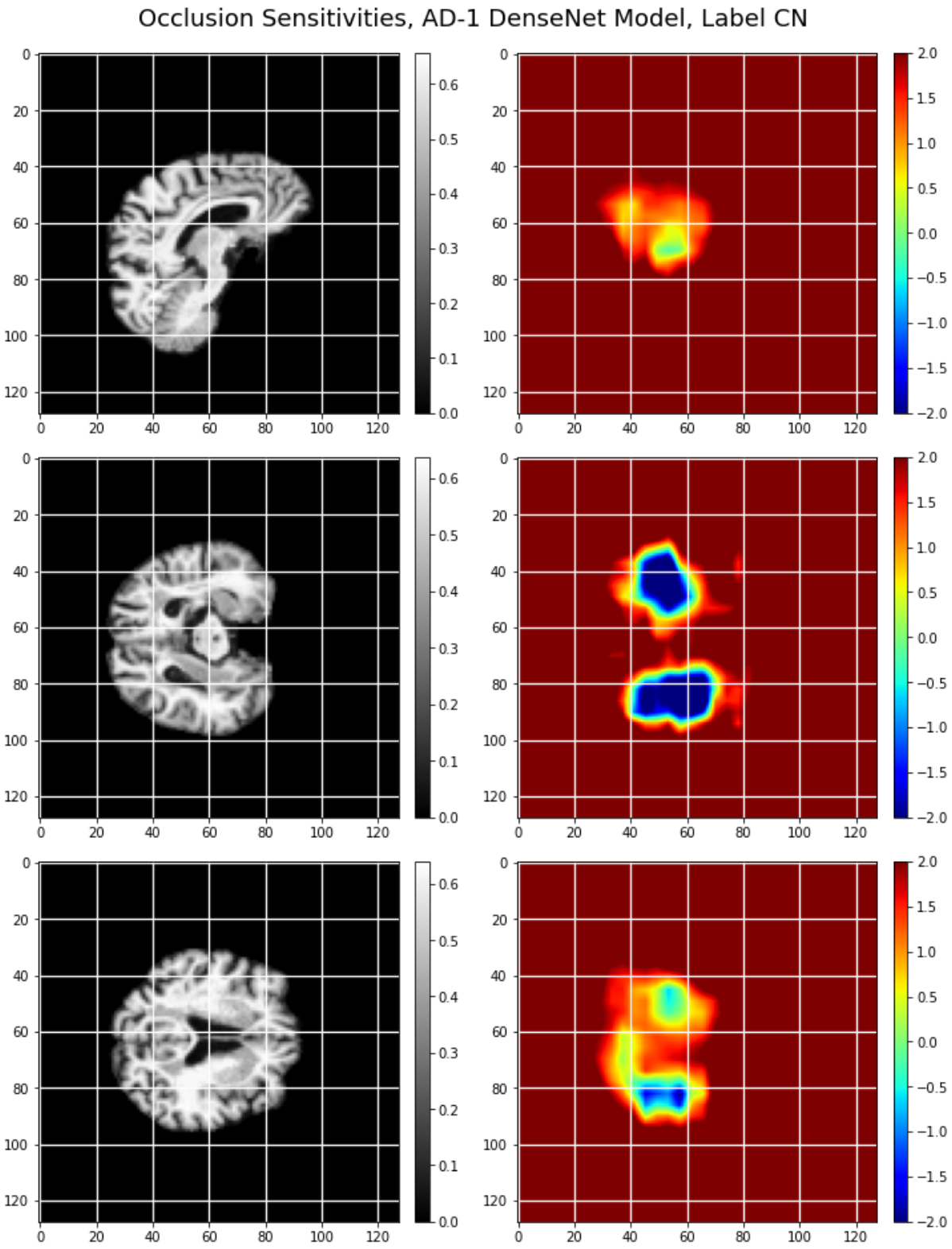


Figure A.2.6: Occlusion sensitivity maps for AD-1 DenseNet model (5-way classification of AD, no pretraining), for prediction of label "CN". To the left is the image slice, to the right is the occlusion map of this image slice. Slices have been made along the following axis (from top): x=55, y=70 and y=55.

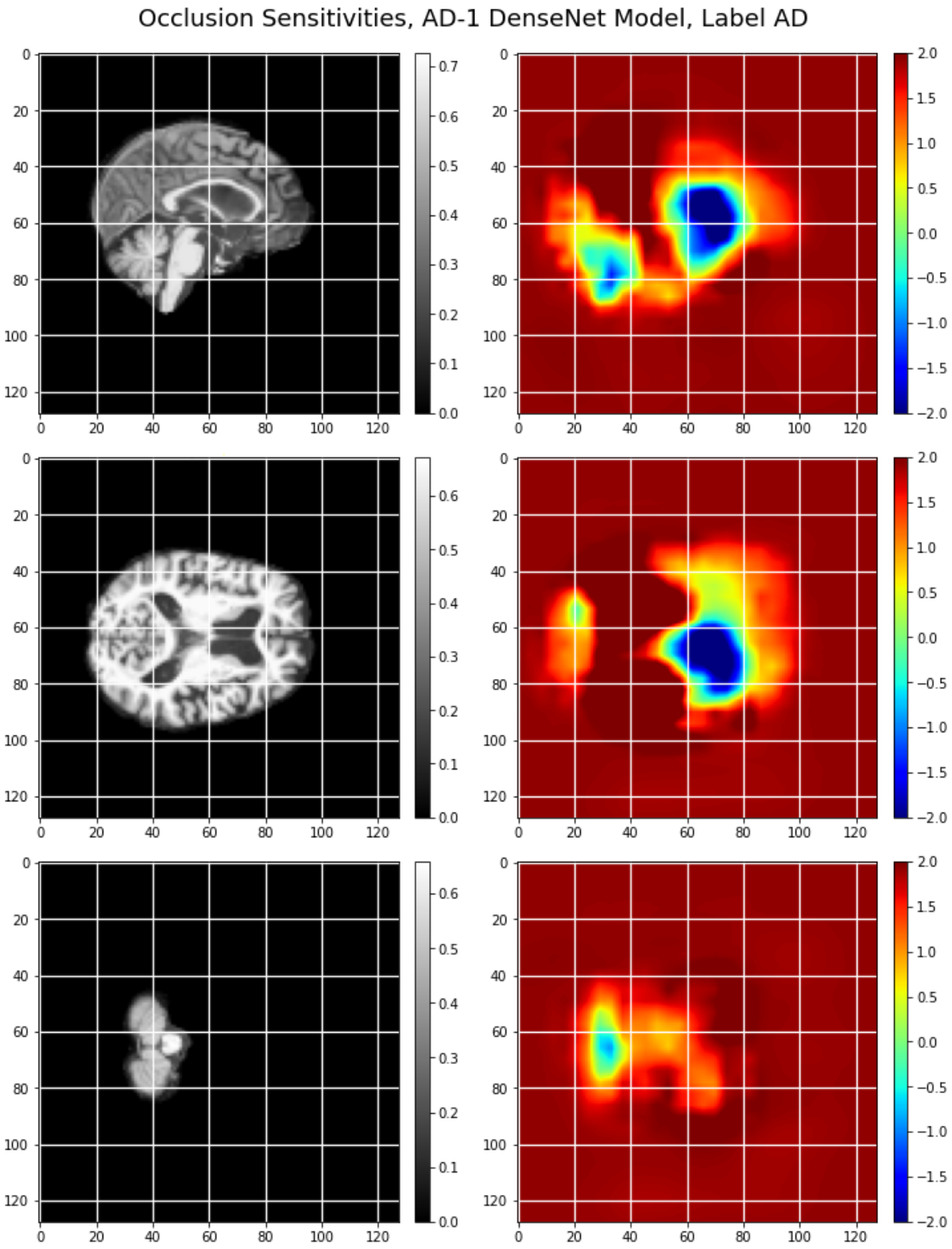


Figure A.2.7: Occlusion sensitivity maps for AD-1 DenseNet model (5-way classification of AD, no pretraining), for prediction of label "AD". To the left is the image slice, to the right is the occlusion map of this image slice. Slices have been made along the following axis (from top): x=60, y=50 and y=80.

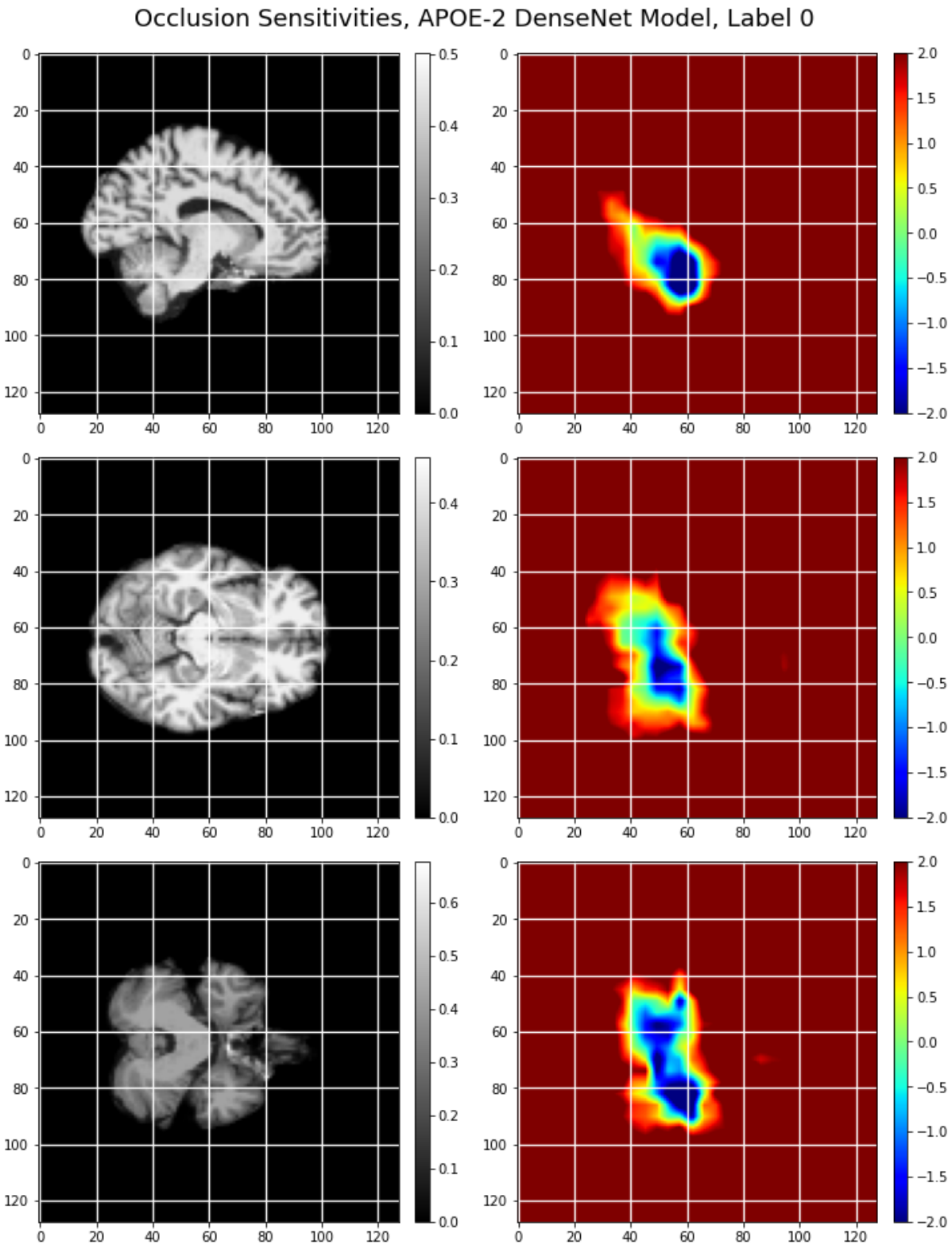


Figure A.2.8: Occlusion sensitivity maps for APOE-2 DenseNet model (binary classification of APOE ε4, pretrained with AD-1 weights), for prediction of label "0" (no alleles). To the left is the image slice, to the right is the occlusion map of this image slice. Slices have been made along the following axis (from top): x=65, y=65 and y=75.

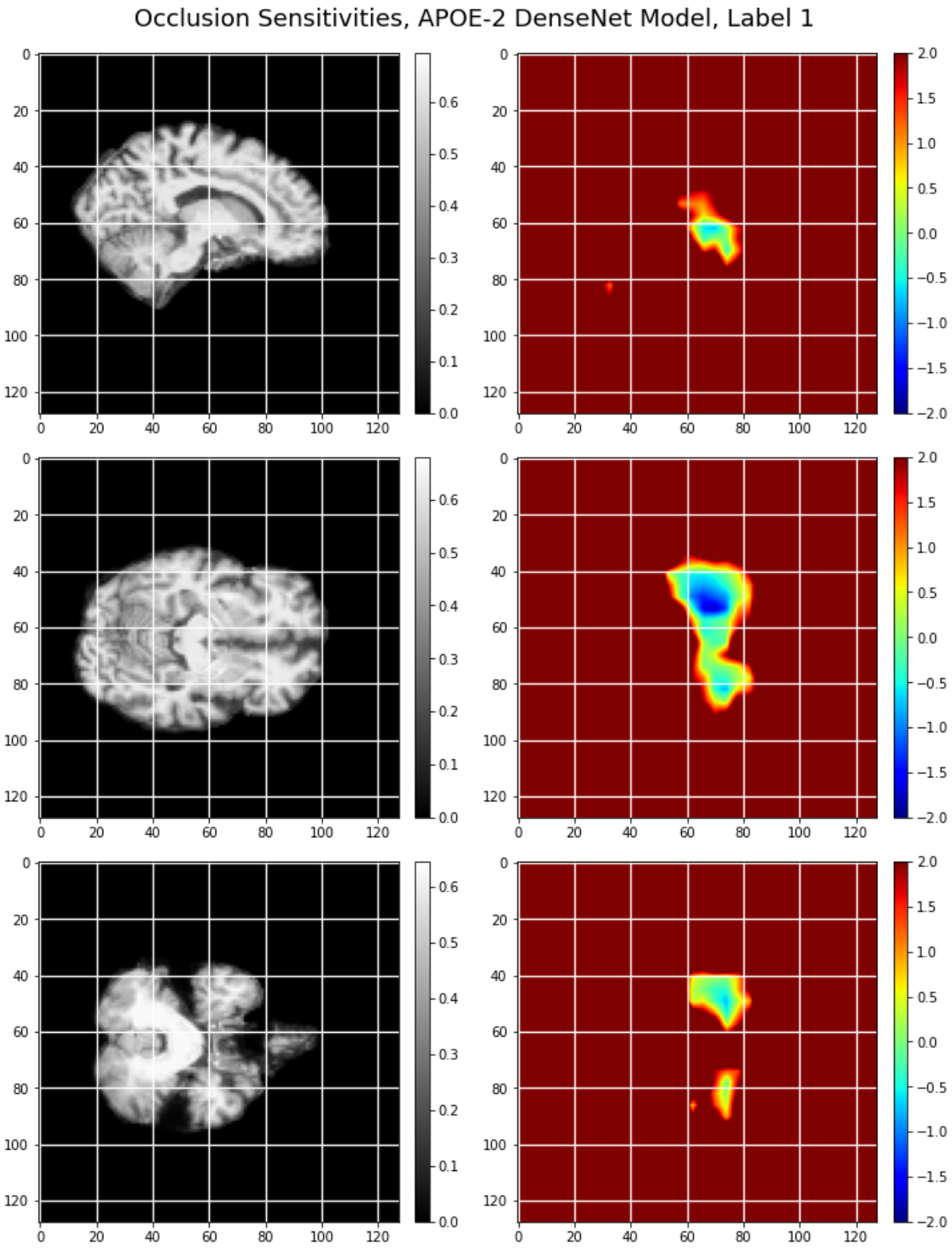


Figure A.2.9: Occlusion sensitivity maps for APOE-2 DenseNet model (binary classification of APOE ε4, pretrained with AD-1 weights), for prediction of label "1" (one or more alleles). To the left is the image slice, to the right is the occlusion map of this image slice. Slices have been made along the following axis (from top): x=55, y=60 and y=70.

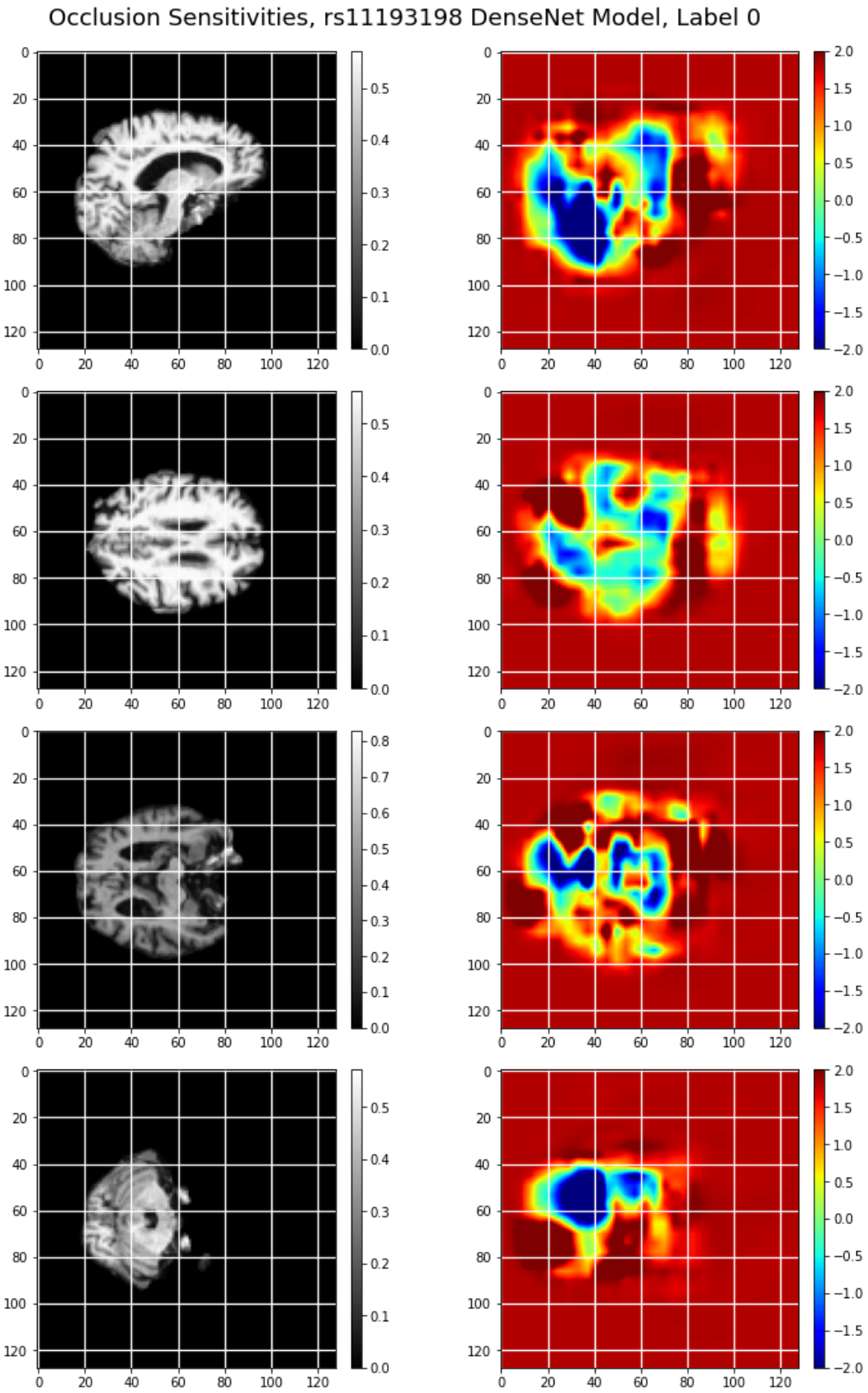


Figure A.2.10: Occlusion sensitivity maps for rs11193198 DenseNet model (3-way classification, pretrained with AD-1 weights), for prediction of label "0". To the left is the image slice, to the right is the occlusion map of this image slice. Slices have been made along the following axis (from top): x=55, y=40, y=60 and y=75.

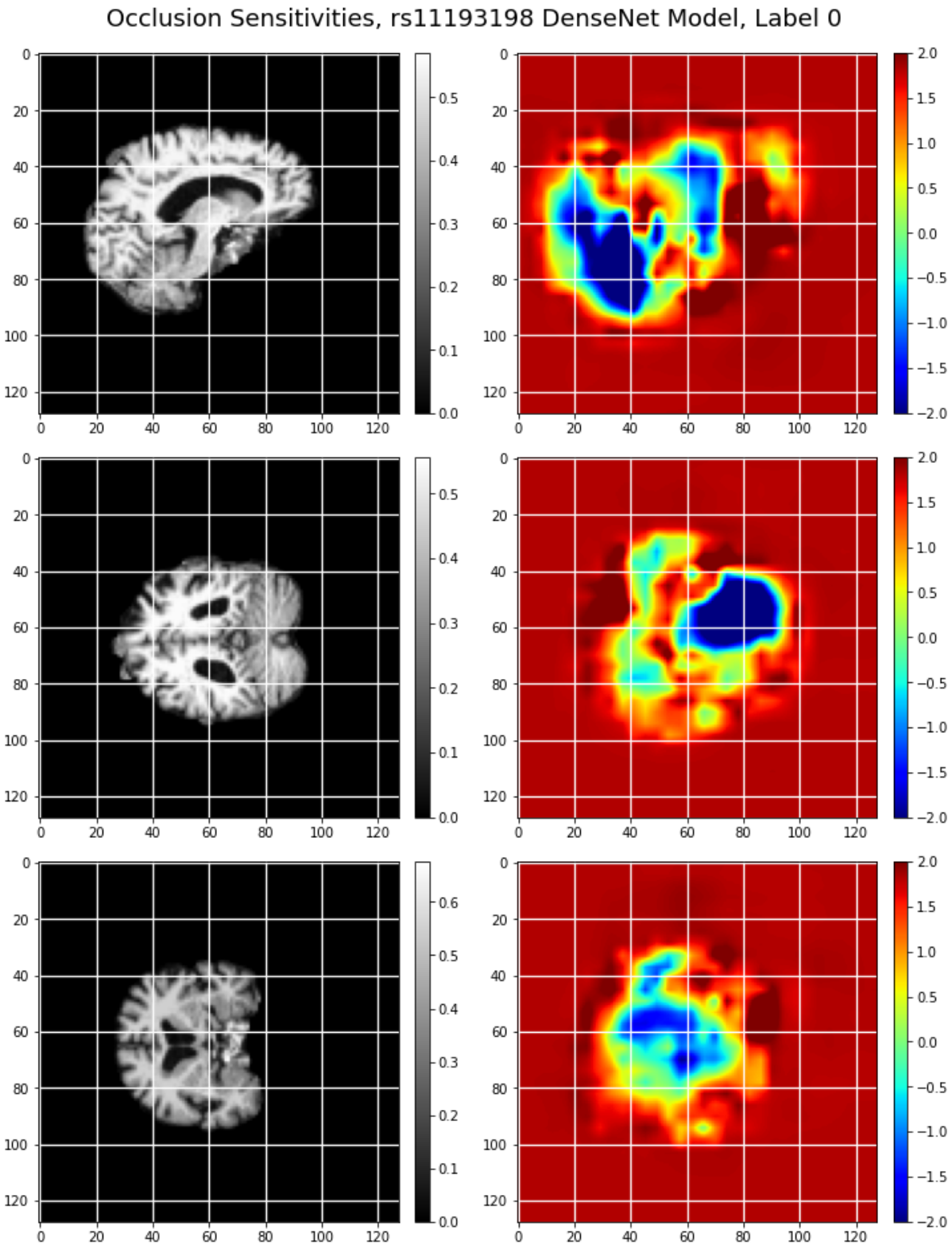


Figure A.2.11: Occlusion sensitivity maps for rs11193198 DenseNet model (3-way classification, pretrained with AD-1 weights), for prediction of label "0". To the left is the image slice, to the right is the occlusion map of this image slice. Slices have been made along the following axis (from top): x=55, z=35 and z=65.

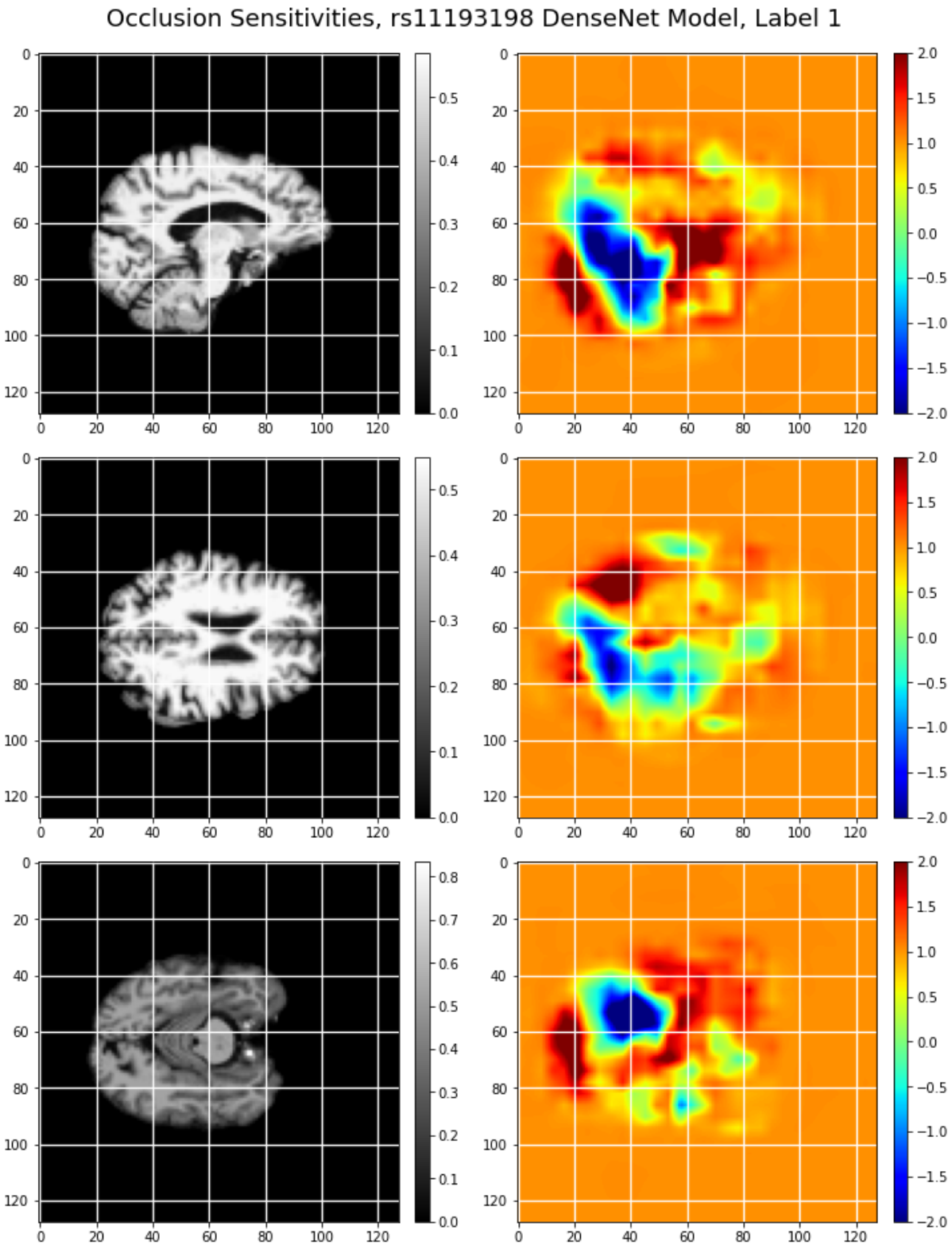


Figure A.2.12: Occlusion sensitivity maps for rs11193198 DenseNet model (3-way classification, pretrained with AD-1 weights), for prediction of label "1". To the left is the image slice, to the right is the occlusion map of this image slice. Slices have been made along the following axis (from top): x=55, y=50 and y=75.

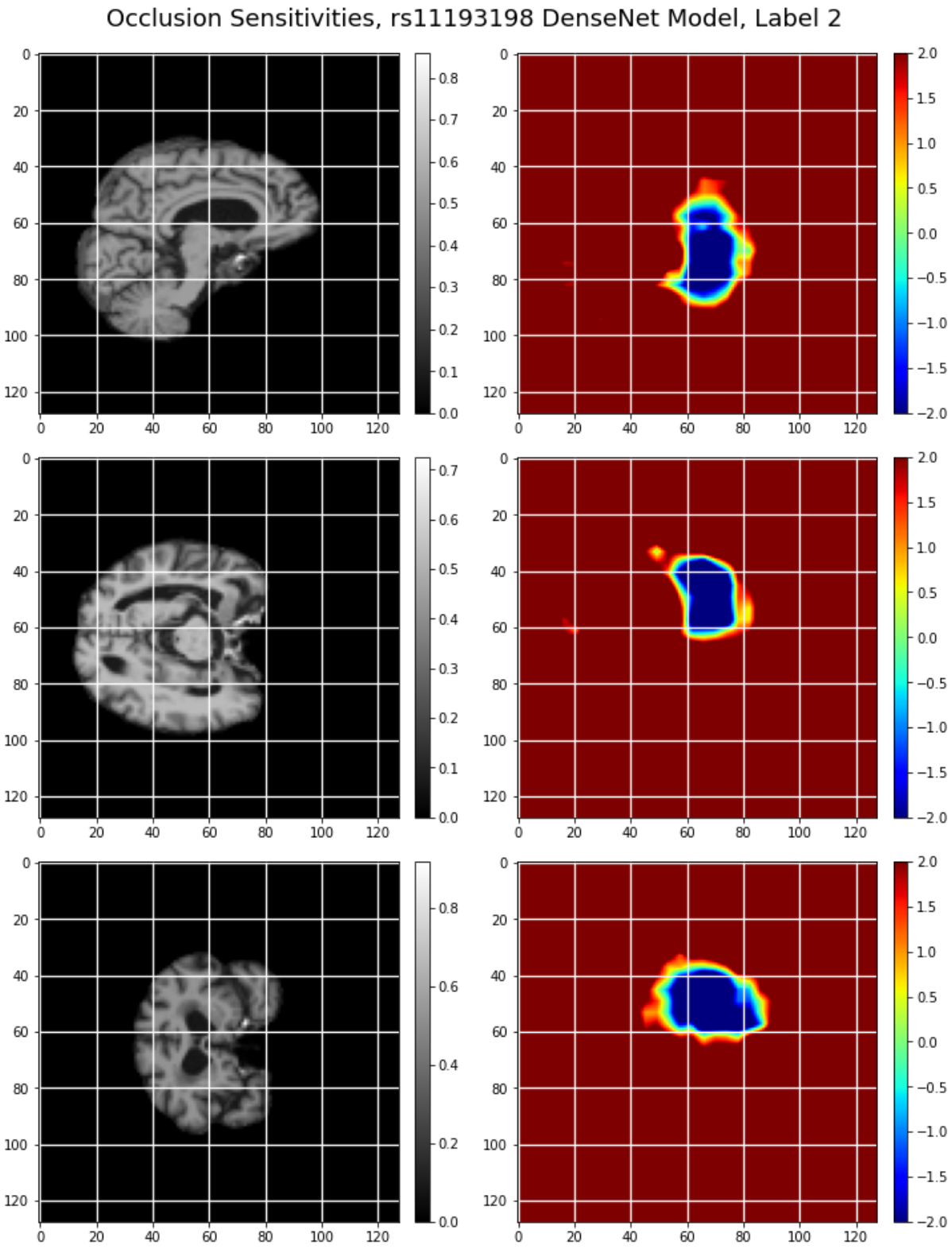


Figure A.2.13: Occlusion sensitivity maps for rs11193198 DenseNet model (3-way classification, pretrained with AD-1 weights), for prediction of label "2". To the left is the image slice, to the right is the occlusion map of this image slice. Slices have been made along the following axis (from top): x=55, y=70 and z=70.

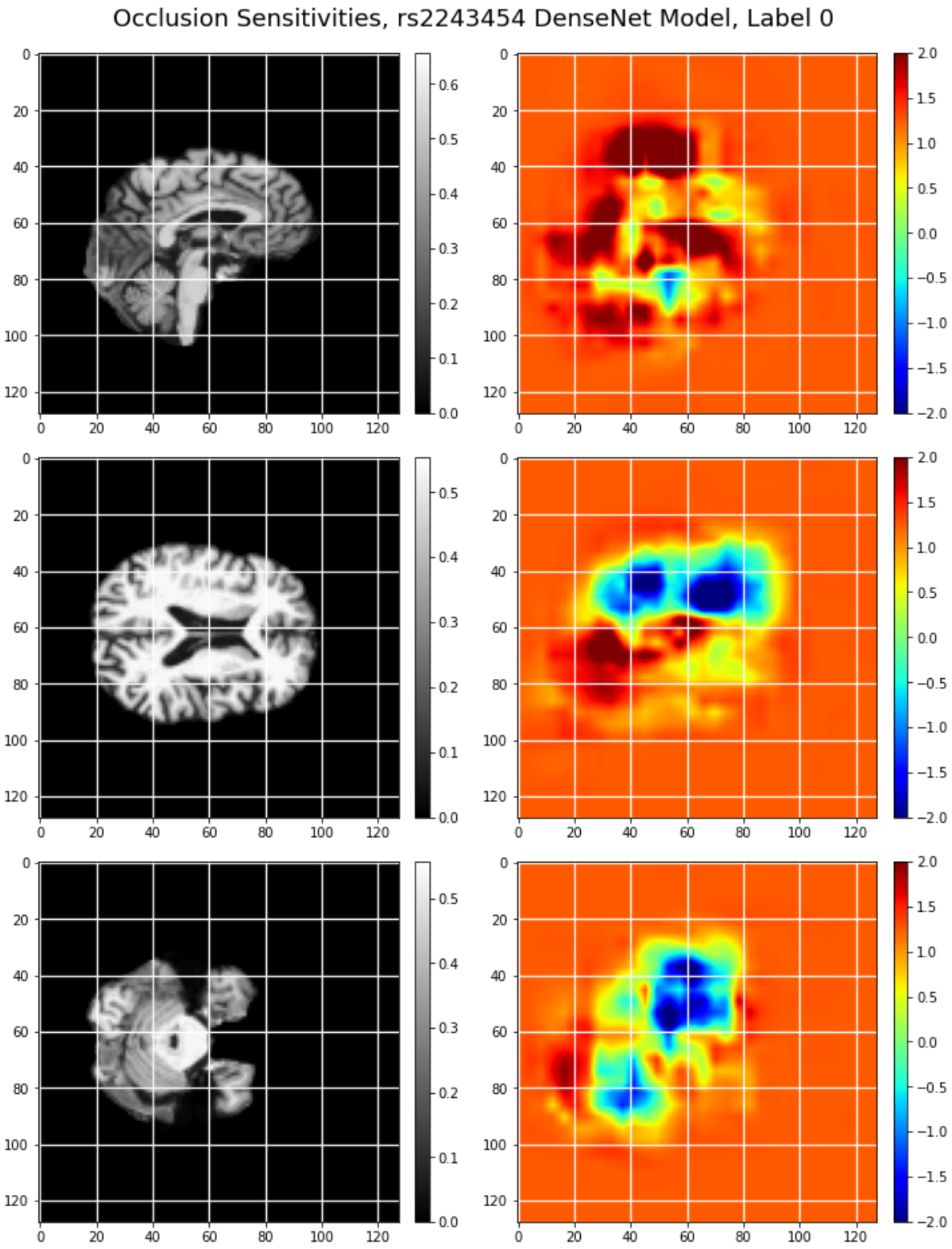


Figure A.2.14: Occlusion sensitivity maps for rs2243454 DenseNet model (3-way classification, pretrained with AD-1 weights), for prediction of label "0". To the left is the image slice, to the right is the occlusion map of this image slice. Slices have been made along the following axis (from top): x=60, y=55 and y=80.

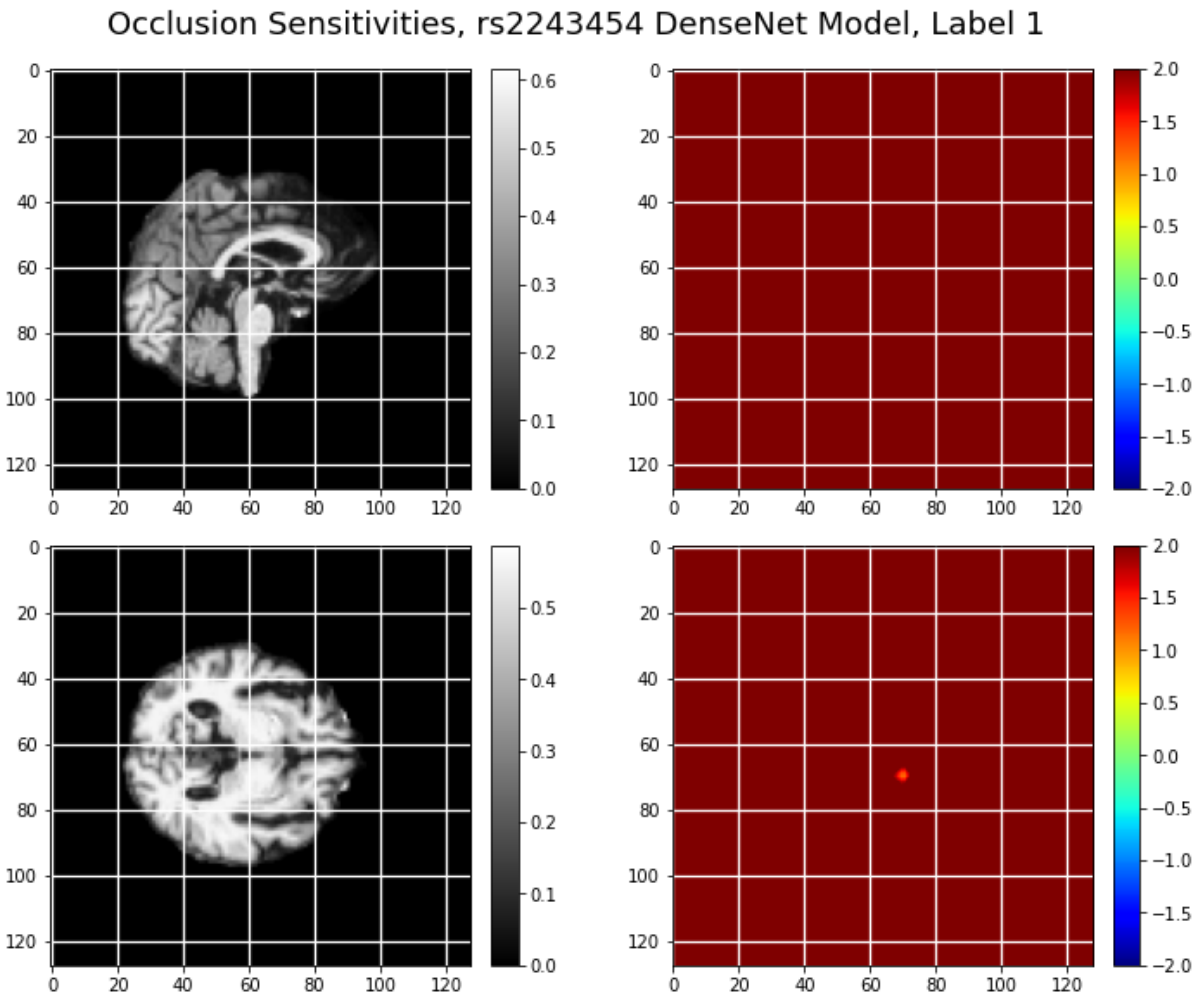


Figure A.2.15: Occlusion sensitivity maps for rs2243454 DenseNet model (3-way classification, pretrained with AD-1 weights), for prediction of label "1". To the left is the image slice, to the right is the occlusion map of this image slice. Slices have been made along the following axis (from top): x=60 and y=60.

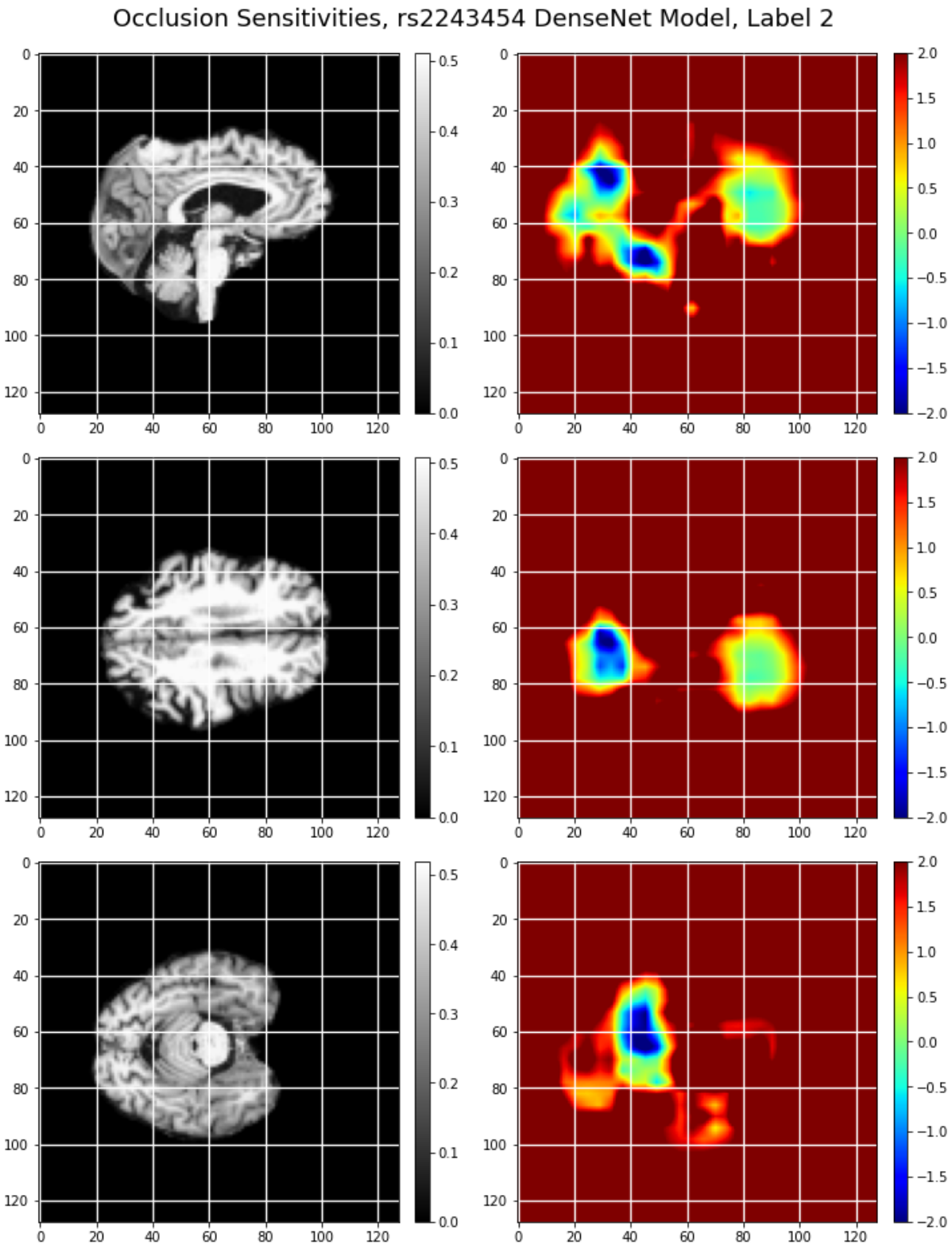


Figure A.2.16: Occlusion sensitivity maps for rs2243454 DenseNet model (3-way classification, pretrained with AD-1 weights), for prediction of label "2". To the left is the image slice, to the right is the occlusion map of this image slice. Slices have been made along the following axis (from top): x=60, y=40 and y=70.

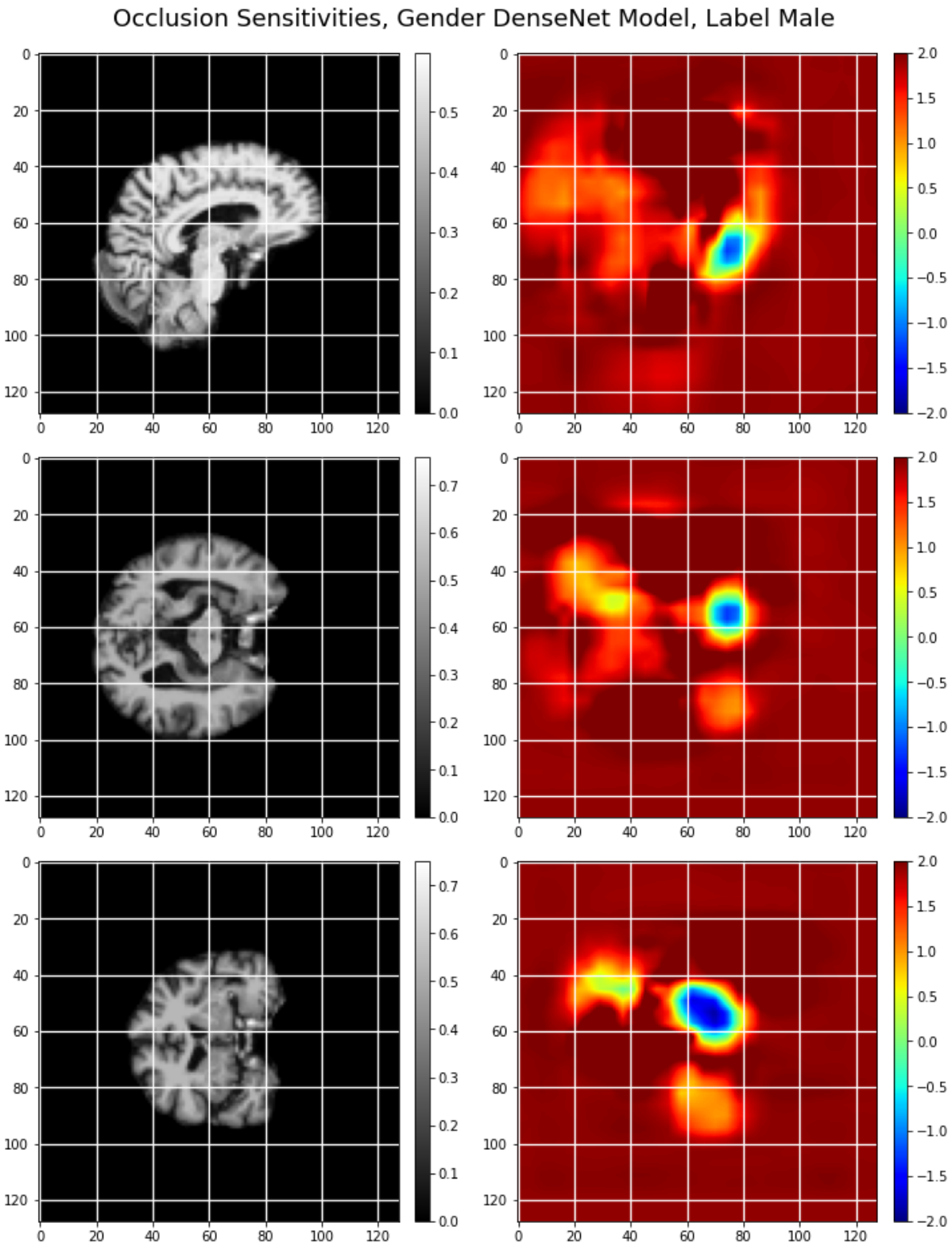


Figure A.2.17: Occlusion sensitivity maps for Gender DenseNet model (binary classification, male/female), for prediction of label "male". To the left is the image slice, to the right is the occlusion map of this image slice. Slices have been made along the following axis (from top): x=55, y=70 and z=70.

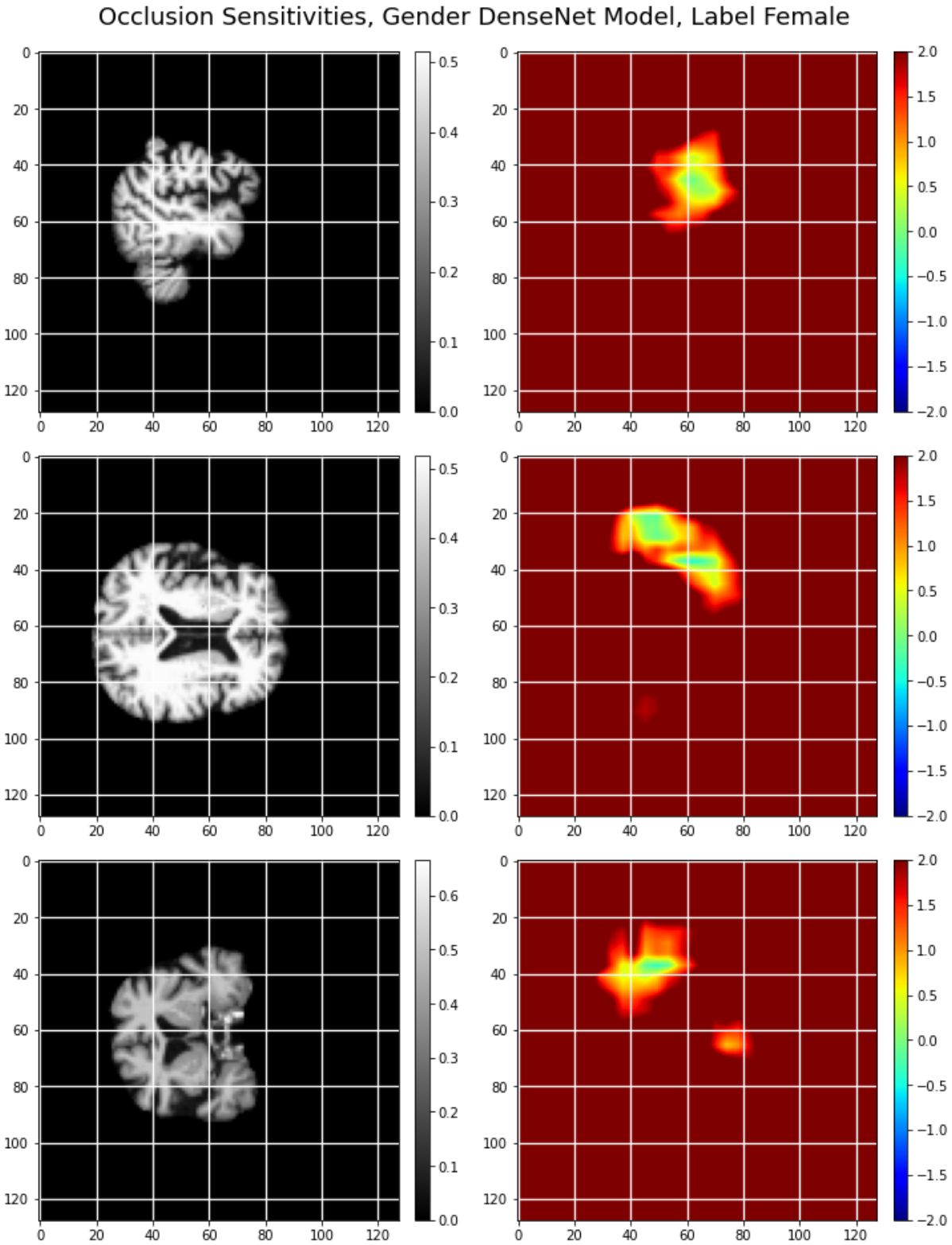


Figure A.2.18: Occlusion sensitivity maps for Gender DenseNet model (binary classification, male/female), for prediction of label "female". To the left is the image slice, to the right is the occlusion map of this image slice. Slices have been made along the following axis (from top): x=35, y=45 and z=60.