An international study on the collection and combination of optimised algorithms for predicting progression in Alzheimer's Disease

MSc Thesis

Lotte E. Mulder
(4379705)

Supervisor Erasmus Medical Center:
Dr. E. E. Bron

Supervisor Delft Technical University:
Dr. F.M. Vos

Biomedical Imaging Group Rotterdam
Erasmus Medical Center
Rotterdam, The Netherlands
2021-05-26

Abstract

Introduction Early detection of Alzheimer's Disease (AD), i.e. before symptom onset, would provide the opportunity for development and testing of interventions at earlier stages, when the disease process may still be altered or interrupted. Computer algorithms combining machine learning with non-invasive imaging and other biomarkers for AD have been developed in an effort to improve early detection methods. However, so far, none of the individual algorithms perform at a level that qualifies for clinical use. In this study, we investigated whether combining several existing AD prediction algorithms improves performance and generalisability.

Methods State-of-the-art AD progression prediction algorithms were collected from the TADPOLE-SHARE project. Algorithms were trained on data from the Alzheimer's Disease Neuroimaging Initiative (ADNI) study and made forecasts of the clinical diagnosis (CN, MCI, or AD). These algorithms were combined using i) simple, unlearned fuser methods and ii) learned fuser methods. In total, seven experiments were conducted, exploring different combination strategies with increasing complexity of fusers. Finally, we implemented and added our own individual algorithm, a residual neural network (ResNet). All individual algorithms and ensembles were evaluated with the multiclass area under the curve (mAUC) and the balanced classification accuracy (BCA) performance metrics. Statistical significance was evaluated with the McNemar test.

Results TADPOLE-SHARE resulted in the collection of eight algorithms, from which five were reused for combination. Overall, combining algorithms slightly improves performance (i.e. increased BCA and mAUC), although improvements were not statistically significant (McNemar test). Both BCA and mAUC showed a trend of improved performance with increasing fuser complexity i.e. data learned fusers and re-entering original data features. DoubleResNet was the best performing ensemble (BCA = 0.809 [$\pm$0.026], mAUC = 0.902 [$\pm$0.020]) and performed slightly better than the best scoring fused algorithm EMCEB (BCA = 0.761 [$\pm$0.029]; mAUC = 0.866 [$\pm$0.020]).

Conclusion These preliminary results suggest that combining pre-existing AD progression prediction algorithms might provide the increase in performance and generalisability needed to enable clinical translation. To do so, future work should be focused on increasing the interoperability of currently existing and newly developed algorithms.

Keywords— Alzheimer's Disease (AD), progression, computer algorithm, ensemble learning, Alzheimer's Disease Neuroimaging Initiative (ADNI), residual network (ResNet)

Contents

1 Introduction

1.1 Alzheimer's Disease

Alzheimer's Disease (AD) is the most common form of dementia, accounting for 50-70% of the dementia cases (Winblad et al., 2016). The most important risk factor for this disease is aging. As a result of the aging world population, the prevalence of AD is increasing dramatically. In 2015 the number of AD cases worldwide was 46 million and, the expectation is that this number will rise to 131.5 million by 2050 (Prince M, Wimo A, Guerchet M, 2015). The symptoms of AD include, amongst others, memory loss, anxiety, and depression. In AD, proteins tau and amyloid-β (Aβ) accumulate in and around the neurons. These proteins play a physiological role in maintaining the cell's morphology and in neuronal growth and repair respectively (Avila et al., 2004). However, in AD, Aβ proteins form plaques outside neurons, interfering with neuron-to-neuron communication. Furthermore, abnormally folded tau proteins form tangles inside neurons, blocking the transport of nutrients and other essential molecules inside neurons (Gaugler et al., 2019). As a consequence, these proteins cause neurodegeneration, leading to the deterioration of cognitive structures and functions associated with AD.

AD is a progressive disease. Its progression comprises three stages during which progressive brain changes can be observed. The three stages range from no or minimal cognitive or behavioral symptoms to severe memory loss and functional impairment and eventually death (Gaugler et al., 2019). The first stage 'pre-clinical AD', is characterised by changes in the brain that are considered biomarkers for AD, e.g., increases in Aβ levels as measured using positron emission tomography (PET) and brain degeneration as demonstrated in magnetic resonance imaging (MRI). In this stage, no or minimal cognitive and behavioural symptoms are notices by the individual. In the second stage (Mild Cognitive Impairment; MCI) individuals can still carry out everyday activities, but noticeable memory loss and cognitive impairment are present. In the final stage (AD) noticeable memory loss and cognitive and behavioral problems impair the ability to carry out everyday activities. Two things should be noted concerning this continuous process. First, not all individuals in the pre-clinical stage convert to MCI, and not all individuals in the MCI stage convert to AD (Ward et al., 2013). Second, the duration of separate stages (and in effect the entire course of the disease) varies between individuals and is dependent on age, genetics, gender, and other factors (Vermunt et al., 2019).

The heterogeneity with which Alzheimer's Disease presents complicates the diagnostic process. On average, dementia is diagnosed 2.8 years after symptom onset in late-onset dementia and 4.4 years after symptom onset in young-onset dementia (development of dementia before age 65) after symptom onset (Van Vliet et al., 2013). There is a large window of opportunity for advancing the diagnostic process, considering that measurable changes of AD start to develop around 20 years before symptom onset (Gordon et al., 2018). Currently, AD is typically diagnosed by a general practitioner or neurologist based on a combination of physical tests, neurological tests, cognitive tests, brain imaging, information on problems in the patient's daily life, their medical history, and other assessments. Criteria for diagnosis are i) impairment of two cognitive domains and ii) difficulties with everyday activities.

Early accurate diagnosis is widely considered critical for the development of new treatments for AD. Despite the decades of extensive research investigating the disease, no curative treatment is available. Clinical trials in which potential medicines are tested are hampered by the lack of accurate early diagnosis (Mehta et al., 2017). Accurate early diagnosis would offer an opportunity to test interventions that may alter or interrupt the progression of the disease.

1.2 Computer algorithms for predicting progression in AD

Computer algorithms can be used as a decision support system for the classification of current and prediction of future clinical status. In this thesis, we focus on the latter, as improving the predictive power of algorithms for AD disease progression may support earlier diagnosis of AD. Computer algorithms utilize machine learning (ML) and biomarkers measurements, enabling the discovery of patterns in the progression of AD from biomarker measurements (Frisoni et al., 2010; Jack et al., 2018). The combination of computer algorithms and biomarkers potentially leads to more accurate and robust diagnoses and predictions in comparison to the earlier described clinical diagnosis protocol.

Numerous promising computer algorithms predicting progression in AD are being developed (Ansart et al., 2021). These algorithms use various prediction methods based on support vector machines (SVM), neural networks (NN), logistic regression, random forests (RF), and other methods. Different features may be used as input. The most frequently used features are T1-weighted MRI, cognition, and socio-demographic features. Less frequently used are fluorodeoxyglucose F^{18} (FDG) PET, APOE, and CSF AD biomarkers.

In general, performance comparisons for different published algorithms in the medical field is challenging because models are often developed and tested using different data sets and/or different metrics. Therefore, despite the reports of many seemingly well-performing, optimised algorithms for AD progression prediction, none are currently used in clinical practice.

1.3 Open science: challenges

Grand challenges (grand-challenge.org) are organised aiming at fairly comparing algorithms with the same task by testing them on the same data and evaluating them with the same performance metrics. Challenges can be interpreted as reviews that capture the state-of-the-art in a particular field. They have proven to be successful in the medical field (Heimann et al., 2009; van Ginneken et al., 2010; Bron et al., 2015; Setio et al., 2017; Marinescu et al., 2018; Bándi et al., 2019; Marinescu et al., 2020). The first grand challenge, SILVER07 on liver segmentation, compared and evaluated 16 algorithms on the same data sets (Heimann et al., 2009). They found that algorithms reporting the highest performance in literature did not perform the best in the grand challenge comparison. A challenge concerning AD is the Alzheimer's Disease Prediction of Longitudinal Evolution (TADPOLE) Challenge (Marinescu et al., 2018). This challenge aimed to make a fair and systematic comparison of 62 models – developed by 34 international research teams – for the prediction of progression in AD. To that end, the challenge used data from the Alzheimer's Disease Neuroimaging Initiative (ADNI) (Weiner et al., 2017; Veitch et al., 2019). The challenge showed that not one single method is best for predicting all biomarkers: clinical status, cognitive decline, and neurodegeneration. In fact, it was found that a different algorithm performed best for each biomarker.

Unfortunately, many algorithms are not directly available for further research. For some algorithms, there is code publically downloadable (e.g. from GitHub), but it is often not trivial to achieve optimal performance on new data due to incomplete information about, or lack of robustness of used models (Hutson, 2018). In 2016, the FAIR guiding principles were established (Wilkinson et al., 2016). The referred paper describes four fundamental principles: Findability, Accessibility, Interoperability, and Reusability. These principles put specific emphasis on enhancing the ability of machines to automatically find and use data, in addition to supporting its reuse by individuals. This paper led to extensive research according to the described principles and boosted open science in academia.

Particularly, the TADPOLE-SHARE project (tadpole-share.github.io), aimed to build a platform for sharing prediction algorithms for the progression in Alzheimer's Disease with the scientific community. It relied on the TADPOLE challenge and targeted to collect optimised algorithms predicting the progression of AD to allow further research as well as evaluation of these algorithms. Participants from the TADPOLE challenge were asked to submit their algorithms to the platform according to the pipeline in figure 2. This setup was selected to enhance the likelihood that submitted algorithms would be replicable, i.e., would yield the same results on reuse, and could thus be further refined, e.g. on different test sets. What is more, (grand-challenge.org) recently added an option to upload algorithms to the website as a Docker container. A docker container is a standalone, executable package of software that includes everything needed to run an application: code, runtime, system tools, system libraries, and settings. So far, 21 algorithms are uploaded as such to the (grand-challenge.org). New initiatives, such as TADPOLE-SHARE and the docker containers on (grand-challenge.org), open up a new world of possibilities.

With the advent of more publicly available databases and extensions of existing databases (Aalten et al., 2014; Ikram et al., 2017) the performance of existing algorithms can be better evaluated, e.g. generalisability can be validated by testing algorithms on external data (Bron et al., 2020). Furthermore, challenges allow for easy and simple combining algorithms that have participated. Indeed, some challenge authors showed that combining algorithms can result in better performance than any of the individual algorithms (Heimann et al., 2009; van Ginneken et al., 2010; Marinescu et al., 2020).

1.4 Ensemble learning

Ensemble learning is a technique that fuses multiple individual models that together decide on the final output. An ensemble can be constructed in two possible canonical topologies: in series and parallel as shown in figure 1. The parallel topology (top part of figure 1) is described most frequently in the literature (Woźniak et al., 2014). In this architecture, each algorithm is independently trained and the outputs of the individual algorithms are combined by a fuser system that makes the final decision.

In the in-series architecture (bottom part of figure 1), the algorithms are ordered or ranked and are applied in sequence. The data points that the first classifier in the sequence predicts falsely are passed on to the next classifier with, for instance, an increased weight compared to the correctly classified data points. As such, the algorithm is tweaked in favour of the data points that the classifiers misclassified earlier in the sequence of classifiers. AdaBoost is an example of an algorithm that has an architecture like the serial topology (Freund and Schapire, 1997). Observe that in this combination architecture, algorithms are trained one by one in a dependent manner.

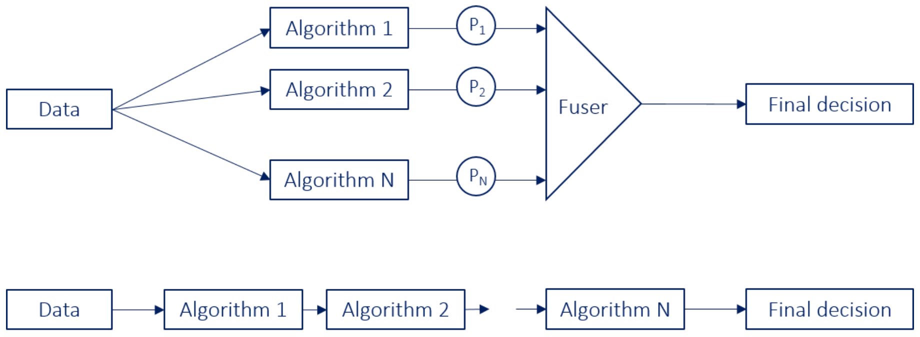


Figure 1: Ensemble of algorithms applied in parallel (top part of the figure) and in series (bottom part of the figure) aiming to achieve enhanced performance.

Previous research has pointed out that an ensemble of *weak* classifiers, i.e. classifiers that are slightly better than random guessing (i.e. accuracy just above 0.5) (Schapire, 1990), results in improved performance compared to individual algorithms (Kittler et al., 1998; Dietterich, 2000; Kuncheva, 2014). Usually, the *weak* classifiers are generated out of one base classifier with e.g bagging, also called bootstrap aggregating, which is a technique that generates multiple classifiers by training one single classifier multiple times on random subsets of the original data set (Breiman, 1996), or using slightly different parameter settings (Hinrichs et al., 2009; Liu et al., 2012; Cheng et al., 2017; Dai et al., 2012). Although not often reported, it has been demonstrated that combining existing, independent, pre-optimised (*strong*) algorithms also improves performance over individual algorithms (Marinescu et al., 2020; van Ginneken et al., 2010; Setio et al., 2017; Codella et al., 2018; Harangi, 2018). These studies combine the individual algorithms in a relatively simple way, e.g. taking the mean or the median of the individual algorithms' outputs. However, algorithms can also be combined by a more complex system that uses the individual algorithms' outputs to train a model and use that model to make a final decision. For example, a meta-algorithm like an artificial neural network (ANN) can combine the individual algorithms by taking into account what it learned from the individual algorithms. Ksieniewicz et al. use a trained classifier to combine weak algorithms with a single-layer perceptron, the most simple form of a neural network (NN) (Ksieniewicz et al., 2018). They report a significant increase in accuracy compared to individual models. A NN is an algorithm that uses a learning technique inspired by the human brain and therewith can model complex patterns and prediction problems. A neural network consists of an input layer, several hidden layers, and an output layer. The layers consist of so-called neurons. A NN can, as any other machine learning model, classify, for example, a patients' diagnosis which can either be CN, MCI or AD. The input data can for example be information on volumes of the ventricles and the age of the patient. The inputs represent a certain value and are entered into the nodes of the input layer. Regular neural networks are characterized by fully connected layers in which each node is connected to all the nodes in the next layer. The determination of the values of the nodes in the next layer, which is called activations, is done as follows. The activation of a node in the next layer is calculated by taking a weighted sum of all the activations in the previous layer, adding a bias term that indicates how much a certain neuron should be activated to have a positive value, and by passing this calculation through an activation function. The weights, which are usually initialized at random, give an indication of how much weight the particular neuron attributes to an input.

In summary, in the training phase, a NN takes examples as input, makes a prediction, learns from its mistakes by backpropagation, and updates its weights to improve itself. A neural network has as its goal, like any other model, to generate a certain output that matches the true label as closely as possible. It reaches this goal by finding the set of weights and hyperparameters that are optimal, decided by the loss function. The loss function represents the performance of the model. Ideally, the loss equals 0, reflecting that the model predicts all examples perfectly.

To the best of my knowledge, there is only one study that uses trained fusers to combine existing, *strong*, independent algorithms. In this study, different computer-aided detection (CAD) systems that participated in grand challenges, are combined in several ways. Amongst others, outputs are combined using a linear discriminant classifier (LDC), a quadratic discriminant classifier (QDC), and a support vector machine (SVM). The reason why there is only one study combining algorithms with trained fusers is that it is a complex problem. First of all, algorithms need to be tested on the same data set to be able to fairly compare their performance and to use their predictions on this data for combination. Furthermore, an ensemble with a trained fuser involves two training and two prediction steps at the levels of individual algorithms and the fuser. Consequently, choices have to be made regarding how the data is used.

The TADPOLE-SHARE project enables combining existing strong algorithms with a trained fuser. Since research has shown that improved performance can be achieved by combining weak classifiers, we hypothesize that this is also true for *strong* algorithms.

1.5 Problem definition

Despite extensive research on AD and its progression, a curative medicine has not yet been found. It is widely believed that noticeable brain damage can be prevented if disease-modifying treatments are administered at an early stage. Computer algorithms have shown good performance predicting the progression of AD, by forecasting future clinical diagnosis Marinescu et al. (2020). However, these algorithms are developed and tested using specific data sets and evaluation parameters, and it is not well known how they would perform in clinical practice. Grand challenges provide a fair and objective comparison of individual algorithms and provide insight into the state-of-the-art performance in a particular field. Moreover, grand challenges offer the opportunity to combine individual models in a relatively easy way, as the models are already evaluated on the same data. Although models show relatively high performance on particular data, their generalisability is often not assessed as they are generally not tested on external data. As such, most of the methods do not have reproducible results.

1.6 This thesis

For this thesis, a validation study will be performed of combinations of individual *strong* algorithms to improve progression prediction in AD.

Aim 1. We will collect existing, optimised algorithms predicting the progression of AD. To that end, we will introduce TADPOLE-SHARE, an initiative that will provide an open-source platform for sharing algorithms for the progression prediction for AD. This project will build upon the TADPOLE challenge, in which 64 algorithms participated.

Aim 2. We will investigate combinations of individual *strong* optimised algorithms for the prediction of progression in AD. Thus far, existing optimised algorithms for progression prediction for AD have not been combined in a more advanced way than by using a mean or median fuser methods. To facilitate an optimal combination of these *strong* algorithms, we propose a framework to combine them. Seven different ensembles will be explored: the mean, the median, and several different NNs.

Aim 3. Lastly, we will compare the performance of ensemble methods and individual methods. Specifically, we will compare 1) individual algorithms and ensembles in general, and 2) unlearned (mean and median) and learned ensembles (NNs).

2 Materials and Methods

2.1 Data

The data used in this study originates from the Alzheimer's Disease Neuroimaging Initiative (ADNI) (adni.loni.usc.edu). ADNI is a longitudinal multicenter study designed to develop clinical, imaging, genetic, and biochemical biomarkers for the early detection and tracking of AD. The main goal of this study is to follow the evolution of AD with the use of biomarkers, together with clinical measures, to assess the brain's structure and function throughout the different disease states. The disease states that were assessed in this thesis are cognitive normal (CN), mild cognitive impairment (MCI), and Alzheimer's Disease (AD). The participants included in the ADNI study are between the ages of 55 and 90 and were recruited at 57 sites in the United States and Canada. Each participant underwent initial tests and baseline measurements. Subsequently, these tests were repeated at intervals of 3 to 12 months. The first ADNI study started in 2004 (ADNI1) and the last ADNI study is still ongoing (ADNI3). The data includes features such as (1) CSF markers of amyloid-beta and tau deposition, (2) various imaging modalities such as magnetic resonance imaging (MRI), positron emission tomography (PET) using several tracers: FDG (hypometabolism), AV45 assessment such as ADAS-COG-13 acquired in the presence of a clinical expert; (4) genetic information such as E4 (APOE4) status extracted from DNA samples; and (5) general demographic information such as age, gender and education level.

In this project, the subsets selected by the organisers of the TADPOLE challenge were used, see Marinescu et al. (2018). These standard ADNI-derived data sets (available via the Laboratory Of NeuroImaging data archive at adni.loni.usc.edu) were pre-processed and contain all participants ever included in any ADNI study. Among these data $D1\_D2$ is a longitudinal data set containing multiple time points of data per subject. Furthermore, data set $D4$ contains data from participants in data set $D1\_D2$ that returned for new measurements after 1 January 2018. Since data set $D1\_D2$ is used for training the algorithms and data set $D4$ for testing, from now on, data sets $D1\_D2$ and $D4$ will be referred to as X_{Train} and X_{Test} for purposes of this thesis. Table 1 shows further details on the data sets.

Table 1: Summary information of data sets X_{Train}, X_{Test}. Converters are participants that are considered to convert to a different diagnosis (CN to MCI, MCI to AD, CN to AD, AD to MCI, MCI to CN, AD to CN) during any of their visits. Data points correspond to time points of data collection (in particular imaging data).

	$\mathbf{X}_{Train}$	$\mathbf{X}_{Test}$
Number of unique subjects	1667	197
Unique data points	12734	234
Follow-up time (mean [std])	0.60 y [± 0.36]	2.6 y [±0.73]
Baseline Diagnosis		
CN	523 (30.2%)	84 (42.6%)
MCI	866 (50.1%)	84 (42.6%)
AD	341 (19.7%)	29 (14.7%)
Converters	468 (27.1%)	33 (16.7%)
Age (mean [std])	73.8 y [± 7.0]	79.57 y [± 7.3]
Gender (%male)	56.6%	55.3%

2.2 TADPOLE-SHARE

2.2.1 Framework

To allow further research and evaluation of the TADPOLE algorithms, participants from the TADPOLE challenge were asked to submit their algorithms to the platform according to the pipeline in figure 2. It is expected that this makes the algorithms easily applicable, as that they can be studied on different test and training data sets. The standardized interface requires users to implement the following functions:

- *train*: this function trains the machine learning model to prepare it for predictions.
- *predict*: a function to make predictions of clinical status, cognitive decline, and neurodegeneration for each month in the following 10 years.
- *evaluate*: function to evaluate the predictions using standardized performance metrics.

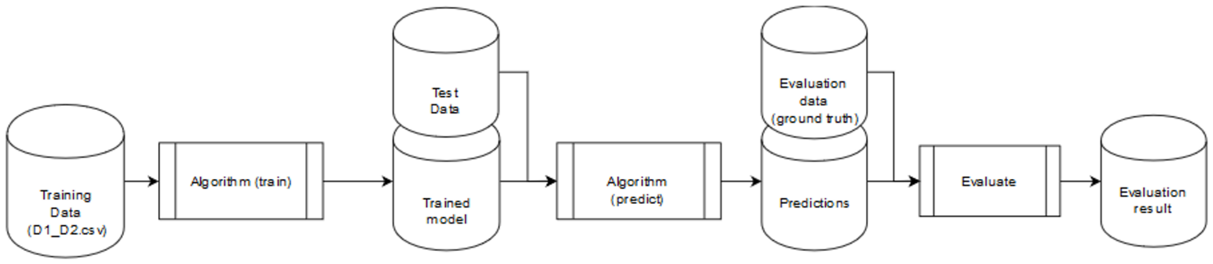


Figure 2: Standardized pipeline algorithms submitted to the TADPOLE-SHARE project

The project was supported with a 2 day (limited due to COVID-19 pandemic) online hackathon that aimed at the submission of rewritten algorithms before the hackathon deadline. Participants' reward for contributing to this project is the co-authorship of the paper that will be published on the TADPOLE-SHARE project in the near future.

2.2.2 Algorithms

Five individual algorithms that participated in the TADPOLE-SHARE project were used in this project: BenchmarkLastVisit, BenchmarkSVM, EMCEB, EMC1, and BORREGOSTECMTY. These algorithms were designed by different research groups. The goal for an algorithm in the TADPOLE challenge was to be the best of all participating algorithms in predicting the clinical status (diagnosis), cognitive decline (ADAS-COG-13), and neurodegeneration (i.e. ventricle volume). In this project, I merely focused on the prediction of clinical status. Unfortunately, the algorithms EMC1 and BORREGOSTECMTY were not submitted in a way that allowed for retraining (despite extensive efforts to facilitate this). As such these algorithms could not be optimized on new data. Short descriptions of the studied algorithms are:

1. **BenchmarkLastVisit (BLV)**
   This algorithm is designed by the organisers of the TADPOLE Challenge to be used as reference algorithms to compare with participant algorithms. It uses the measurements of each target from the last available clinical visit as the forecast. Features selected for this algorithm are diagnosis, ventricle volume, and ADAS-Cog13.

2. **BenchmarkSVM (BSVM)**
   BenchmarkSVM is also designed by the organisers of the TADPOLE Challenge to be used as a reference algorithm. It uses an out-of-the-box support vector machine (SVM) classifier and regressor to provide a forecast. Selected features for this algorithm are diagnosis, age, ADAS-Cog13, ventricle volume, intracranial volume (ICV), and APOE gene (playing an important role in AD).

3. **EMCEB**
   The algorithm EMC-EB also uses SVMs. Feature selection was done automatically resulting in 204 features.

4. **EMC1**
   The algorithm EMC1 uses novel event-based modeling (EBM) technique, which is shown to be more accurate than existing state-of-the-art EBM methods (Venkatraghavan et al., 2019). Feature selection was done automatically resulting in 250 features.

5. **BORREGOSTECMTY (BTMTY)**
   The BORREGOTECMTY algorithm applies an ensemble of 50 linear regression models, which are weak classifiers. Feature selection was done automatically resulting in 500 features.

Table 2: Summary of methodology used by the individual algorithms to predict the clinical status. Keywords: SVM - Support Vector Machine, DPM - disease progression model, BCA - balanced classification accuracy, mAUC - multi-class area under the curve

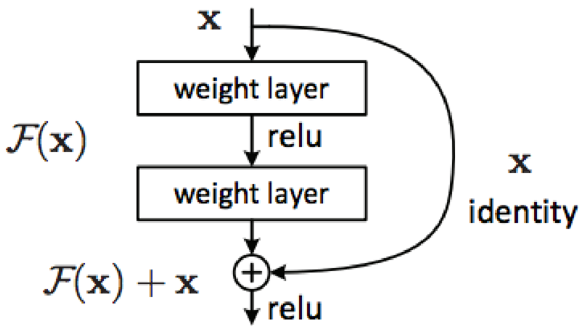
Algorithm	**Feature Selection**	**Number of features**	**Diagnosis prediction model**	**Performance in TADPOLE Challenge**
BenchmarkLastVisit	None	3	Constant model	BCA: 0.774, mAUC: 0.792
BenchmarkSVM	Manual	6	SVM	BCA: 0.764, mAUC: 0.836
EMC-EB	Automatic	204	SVM	BCA: 0.805, mAUC: 0.907
EMC1	Automatic	250	DPM + 2D spline + SVM	BCA: 0.811, mAUC: 0.898
BORREGOSTECMTY	Automatic	500	Regression ensemble	BCA: 0.808, mAUC: 0.866

Table 2 summarizes some aspects of the algorithms. More information on these algorithms can be found in (Marinescu et al., 2020). Observe that the potential benefit of combining them derives from the use of different classification strategies as well as relying on different information to do so (ranging from brain imaging, genetic information, and cognitive tests information).

2.3 ResNet algorithm

Since algorithms EMC1 and BTMTY could not be optimised on new data, TADPOLE-SHARE resulted in three retrainable and retestable algorithms. However, it is doubtful if the algorithms BLV and BSVM can be considered *strong* as they are benchmark models and were not carefully optimised. Tabel 1 demonstrated that the benchmark models perform worse in comparison to e.g. algorithm EMCEB. We, therefore, developed our own model: a ResNet.

Residual networks (ResNets) were first introduced by (He et al., 2016). This journal paper became one of the most influential papers in modern deep learning. ResNets make use of skip connections. As the name suggests, layers can be skipped in the neural network providing an alternative path for the gradient. This methodology is experimentally validated to be beneficial for the models' convergence. Figure 3 shows the fundamental building block of the residual neural network proposed in (He et al., 2016). The input, x, is first fed through a fully connected layer followed by a ReLu activation. Thereafter, x is fed through a fully connected layer resulting in $F(x)$. Concurrently, the input x skips the fully connected layer and is added to $F(x)$. Finally, $F(x)$ + x is activated by the ReLu activation function. These types of networks are a standard module in many convolutional architectures, however, they can be used in any type of NN. This type of framework eases the training of networks that are substantially deeper than those used in previous research.



x
weight layer
$\mathcal{F}(\mathbf{x})$
relu
weight layer
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

Figure 3: The fundamental building block of a residual network

We explored to what extent our developed network can predict the progression of AD on X_{Test} in comparison to previously introduced methods. The models' architecture is presented in figure 4. The input features are passed through a fully connected (fc) layer with fifty nodes. Activation is determined by the relu activation function. Thereafter, ten residual blocks (figure 3) are embedded in series. Finally, the input is downscaled from fifty nodes to the three classes and activated by the softmax function.

The model was trained and validated on both the data split scenarios (explained in section 2.5.3). Feature selection methodology was adopted from algorithm EMCEB resulting in 203 features. Missing values in the data set were filled with forward filling followed by backward filling per RID. The remaining missing values were set to -0.05.

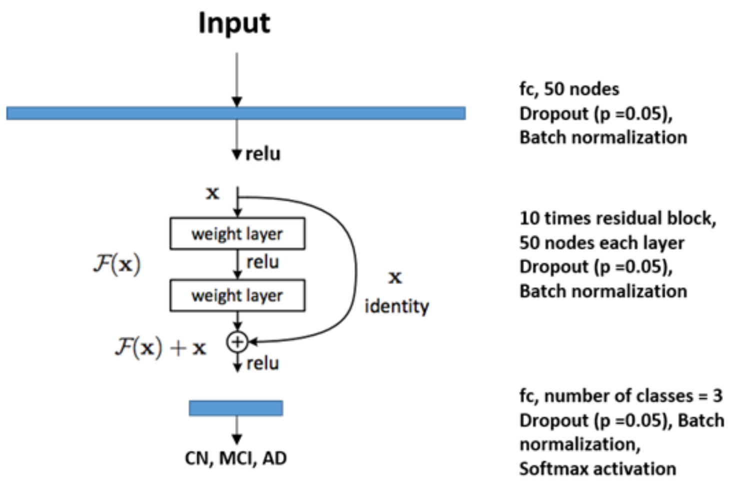


Figure 4: The ResNet model architecture. Observe that the residual block was embedded in series for ten times. Each layer in the model is activated by the relu activation function except for the final layer: this layer was activated by the SoftMax function.

Hyperparameter optimisation was performed manually on the validation set. Several different setups were evaluated (see Appendix B) and the final hyperparameter setup was selected based on the the BCA score on the validation set. Batch normalization and dropout techniques were deployed. During the training process, the learning rate (LR) was optimised by *'reducing the LR* *on plateau'*. This is a widely used adaptive LR scheduler and exists as a function, amongst others, in Pytorch (Paszke et al., 2019). This function divides the LR by 2-10 if the BCA score on the validation data is not improving for a certain number of epochs. We divide the LR by 2 if it is not improving for 5 epochs. The training process automatically terminates when the LR reached the minimal value 1e-5. The model in the epoch with the highest BCA score is saved, and loaded when the LR gets updated.

Note that this algorithms is retrainable, however, was not retested yet and could therefore no be used as input for learned fusers.

2.4 Ensemble Methods

Two general frameworks are described to construct ensembles (combinations) of algorithms with i) an un-trained fuser and ii) a trained fuser. Next, the fuser methods and the conducted experiment setups are explained. Finally, the performance evaluation metrics are explained.

2.4.1 General frameworks to combine algorithms

First, Figure 5 introduces a general framework for ensemble learning with an unlearned fuser (e.g. applying mere averaging of predictions). This framework is relatively simple as the outputs are directly fused to give a final decision. In step 1, the individual algorithms are trained on dataset X_{Train}. In step 2, predictions are made with the trained individual algorithms on data set X_{Test}. The outputs are then fused with the (untrained) fuser, giving the final decisions.

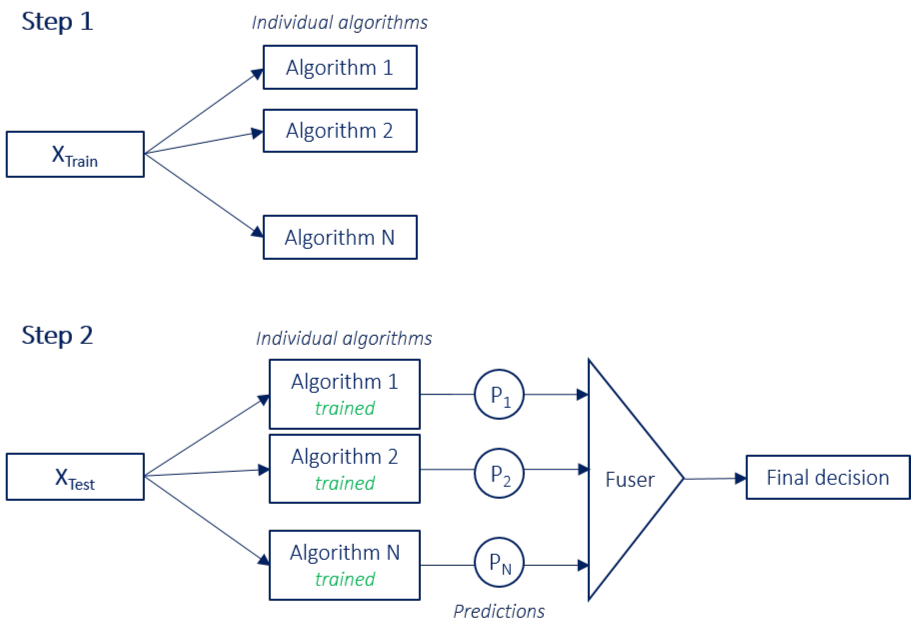


Figure 5: Ensemble framework for combining algorithms with an untrained fuser: step 1) train individual algorithms on data set X_{Train}, step 2) make predictions with individual algorithms on data set X_{Test} and fuse the predictions of the individual algorithms to give the final decision

Alternatively, in figure 6, the general framework for ensemble learning with a trained fuser is shown. This framework involves 3 steps and is more complex than the previously introduced framework. In step 1, the individual algorithms are trained as before (applying X^{Ind}_{Train}). In step 2, the trained individual algorithms make predictions on data set X^{Ind}_{Test} that are subsequently used as inputs for training the fuser. Finally, in step 3, predictions on data set X_{Test} are made and combined by the trained fuser to yield the final decision.

While doing so, an important issue is: *How should one generate X^{Ind}_{Train} and X^{Ind}_{Test} in this framework*?

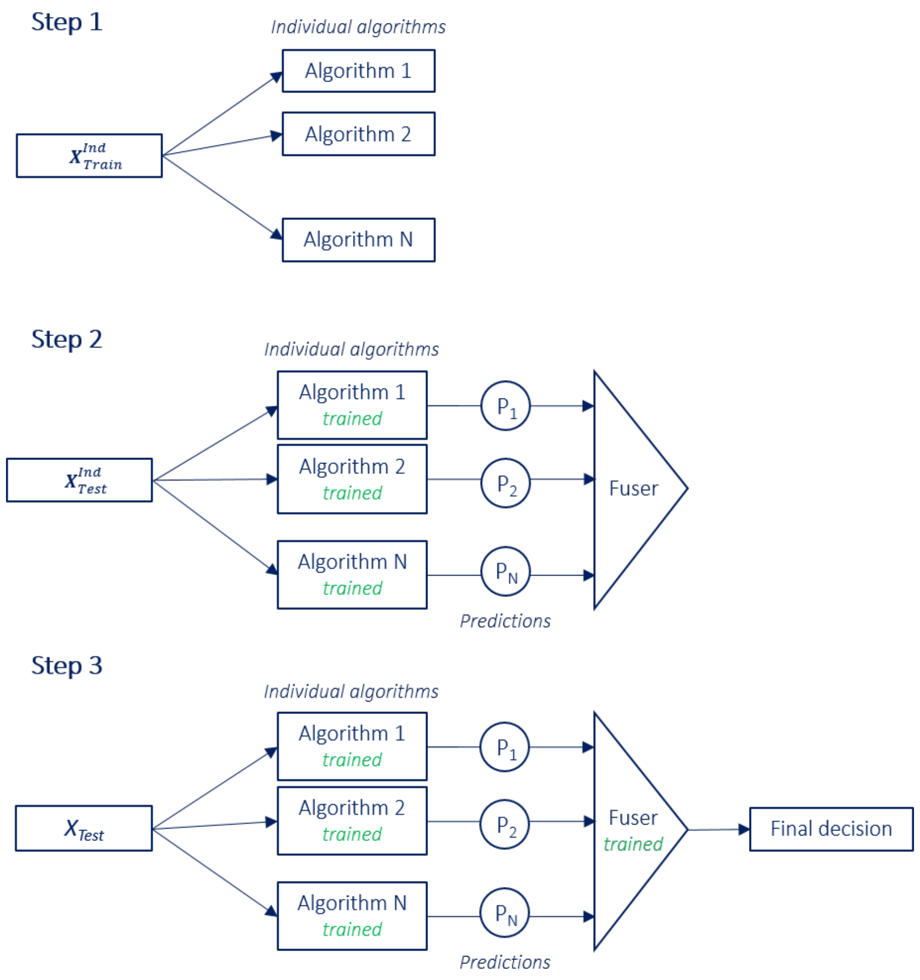


Figure 6: Ensemble framework for combining algorithms with a trained fuser.

Two scenarios are applied to define data sets X^{Ind}_{Train} and X^{Ind}_{Test}:

- Single training set scenario
  Essentially, data set X_{Train} is not split in this scenario. As such, the individual algorithms are trained on the complete set X_{Train} in step 1. Subsequently, in step 2, the same data set X_{Train} is used to train the fuser. A possible downside of this scenario, however, is that it might result in overfitting.

- Separated training sets scenario

  A more robust way would be to perform a k-fold cross-validation. Here, the dataset X_{Train} is split up into k-groups. k-1 groups are used to train the individual algorithms and the remaining group is used for validation. This is repeated so that each group is used once for validation. In this way, independent predictions are made on the entire data set X_{Train}. However, since some of the algorithms are complex and take a long time to train, this scenario was found too time-consuming.

  Instead, an experiment was performed with separated training and test sets: in this scenario, data set X_{Train} is split in half so that half of the data, X_{Train}^{Ind}, is used to train the individual algorithms in step 1 and the other half, X_{Test}^{Ind}, for training the fuser in step 2. A positive aspect of this scenario is that different data is used in steps 1 and 2. However, a disadvantage is that the individual algorithms are trained on part of the data. This could results in weaker individual algorithms, in particular with smaller datasets.

2.4.2 Fusers

In this project, three different ensemble methods were used.

First, the algorithms were combined in several setups by merely averaging the predictions. This model could be expected to give a slightly improved result as some models might be positively biased regarding their predictions, while others might be negatively biased (Marinescu et al., 2018).

Second, the algorithms were combined in several combinations by selecting the median probability. For this ensemble method, a similar result is expected as for ensemble method 1, but it might be less sensitive to outliers.

The last more complex combination method is applying a NN (explained in section 1.3). This is a method that must be trained as in framework 2 (figure 6). Notice that while applying such a combination method, only the three retrainable individual algorithms (BLV, BSVM, EMCEB) can be deployed.

Accordingly, I will separately study (1) the situation in which all six algorithms are applied without retraining while only combining with the unlearned methods; (2) the situation in which three individual retrainable algorithms BLV, BSVM, and EMCEB are retrained and subsequently the combining classifier (see figure 6).

2.4.3 Data split

The total data was split three times, as illustrated in figure 7. Initially, $Split_1$ splits the total data set in data set X_{Train} and X_{Test}. Notice that X_{Test} includes data from the patients in X_{Train}, but at a later stage. In particular, it separates the (before January 1, 2018) time points of all RIDs (patient IDs) from newer time points (after January 1, 2018) of participants enrolled in the study. This split was originally applied in the TADPOLE challenge so that at the time of submission of the algorithms, X_{Test} did not yet exist. Therefore, it was not used in the development of the algorithms. Moreover, this split mimicked a clinical setting in which patients are to be tracked over time. However, this initial split complicates our machine learning strategies for several reasons. First, the class balance in X_{Test} is not similar to the class balance in data set X_{Train} (see table 1). Moreover, there might be a bias as algorithms were developed using data from the same patients that are also in the test set. Also, the test data set is small in comparison to X_{Train} (table 1).

$Split_2$ splits the subjects in data set X_{Train} into two sets: one for training the individual algorithms, and the other for testing them: X^{Ind}_{Train} and X^{Ind}_{Test} respectively. This was done in such a way that, subset X^{Ind}_{Train} and X^{Ind}_{Test} do not have overlapping participants. In other words, all time points belonging to the same subject are put in the same set. The latter step is only necessary, however, for the experiments that use a learned fuser. As already stated above, the ratio of data points between X^{Ind}_{Train} and X^{Ind}_{Test} was set to 1:1 (50%/50%)

$Split_3$ is, like $Split_2$, only applied for the framework with the neural networks to have training data and validation data: X^{NN}_{Train} and X^{NN}_{Val} respectively. Both $Split_2$ and $Split_3$ are random and stratified based on their baseline diagnosis and on if they are converting to another diagnosis throughout their visits.

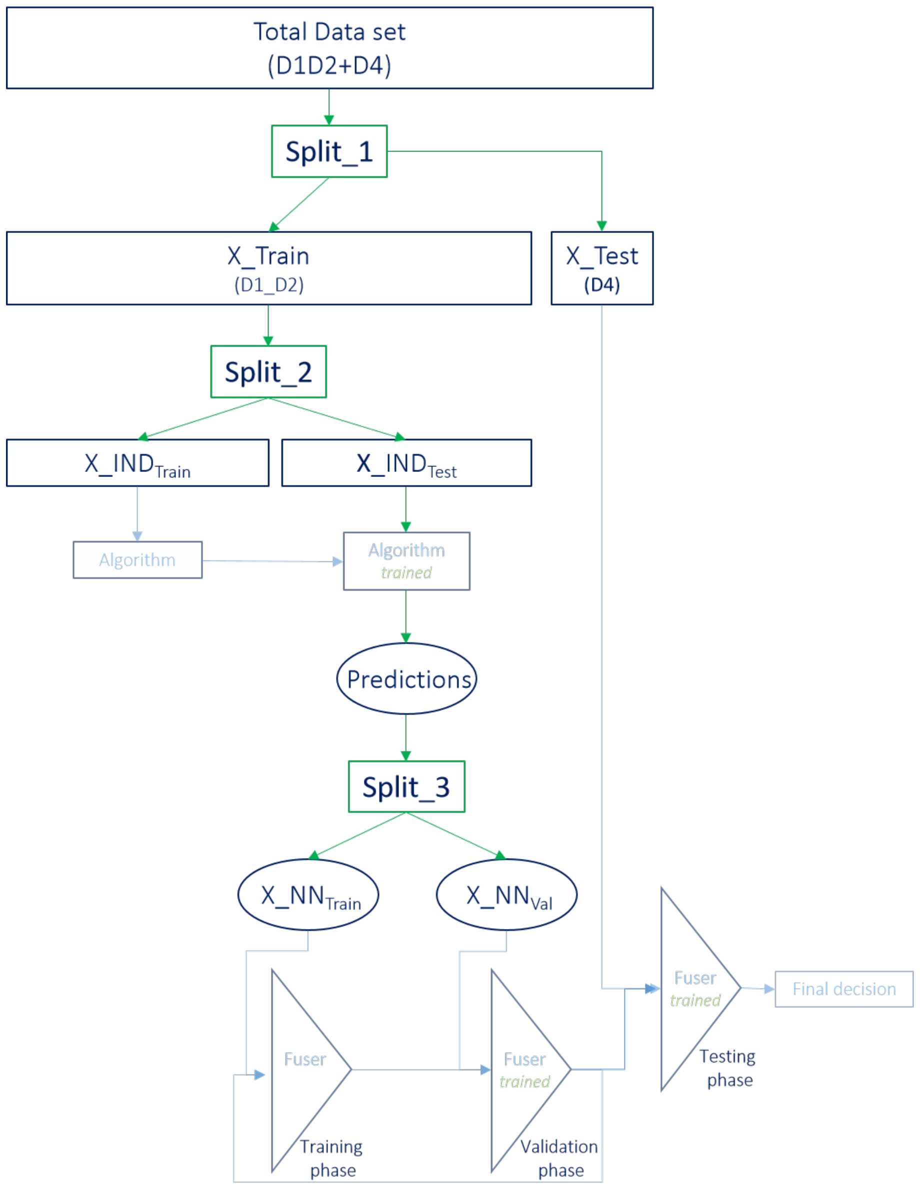


Figure 7: Illustration of how the data is split for the ensemble with an learned fuser method. In total, the data is split three times. Split$_1$: the total data is split in X_{Train} and X_{Test}, Split$_2$: X_{Train} is split in X^{Ind}_{Train} and X^{Ind}_{Test}

2.5 Experimental setup

Seven experiments were conducted as described below. Each individual algorithm yields three inputs to the fuser: the relative probability of CN, the relative probability of MCI, the relative probability of AD. Experiments six and seven also re-enter the union of the inputs of the fused algorithms (203 features) as additional information. Table 3 summarises the seven conducted experiments.

Table 3: Summary of the seven conducted experiments. Notice that experiment 1 and 2 are conducted twice: in the single and separate training set scenario. The input algorithms EMC1 and BTMTY were only used as input in the single training set scenario as they are not retrainable. Experiments 1b and 2b were conducted with and without fusing ResNet. Results without fusing ResNet are used for direct comparison with learned ensembles. Results with fusing ResNet are presented in Appendix A.

	Fuser	**Unlearned/ Learned**	**Scenario**	**Input**
Experiment 1a	Mean	Unlearned	Single training set	BLV, BSVM, EMCEB, EMC1, BTMTY, ResNet
Experiment 1b	Mean	Unlearned	Separate training set	BLV, BSVM, EMCEB, (ResNet)
Experiment 2a	Median	Unlearned	Single training set	BLV, BSVM, EMCEB, EMC1, BTMTY, ResNet
Experiment 2b	Median	Unlearned	Separate training set	BLV, BSVM, EMCEB, (ResNet)
Experiment 3	OneNeuron	Learned	Separate training set	BLV, BSVM, EMCEB,
Experiment 4	TwoLayers	Learned	Separate training set	BLV, BSVM, EMCEB,
Experiment 5	ResNetFuser1	Learned	Separate training set	BLV, BSVM, EMCEB,
Experiment 6	ResNetFuser2	Learned	Separate training set	BLV, BSVM, EMCEB and 203 features
Experiment 7	DoubleResNet	Learned	Separate training set	BLV, BSVM, EMCEB and 203 features

2.5.1 Unlearned ensembles

In experiments 1 and 2, individual algorithms were combined by taking the mean and median of the output probabilities, respectively. These experiments were both conducted twice: in the single and separate training set scenario. First, the five algorithms that participated in TADPOLE-SHARE (BLV, BSVM, EMCEB, EMC1, and BORREGOSTECMTY) plus the developed ResNet were trained according to the single training set scenario and subsequently combined. Second, the four retrainable algorithms (BLV, BSVM, EMCEB, and ResNet) were trained according to the separate training set scenario and subsequently combined.

All possible combinations of the algorithms were investigated resulting in 57 experiments for both mean and median in the single training set scenario and 11 experiments for both mean and median in the separate training set scenario. In total 57x2 + 11*2 = 136 experiments were conducted.

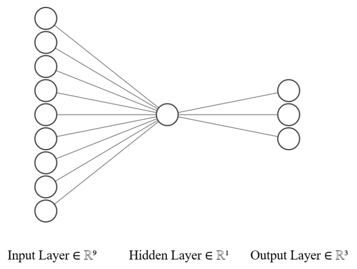
2.5.2 Learned ensembles

The experiments with the learned ensembles were conducted only in the separated training sets scenario explained before. First, the three trainable algorithms (BLV, BSVM, and EMCEB), were combined with relatively simple NNs.

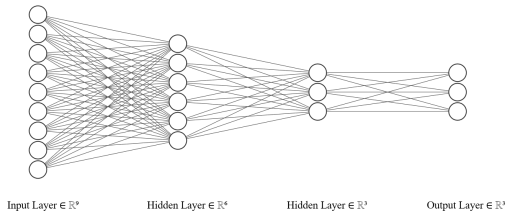
In experiment 3, the NN architecture has one hidden layer with a single node (figure 8a). We refer to this ensemble as *OneNeuron*. This experiment was conducted to study how the NN performs in comparison to the unlearned mean. We hypothesize that this method performs at least as well (as it may just compute the mean of its inputs).

In the fourth experiment, a NN was used that has a somewhat more complex architecture: two hidden layers, with six and three nodes (figure 8b). This ensemble is referred to as *TwoLayers*.

Both experiment 3 and 4 use an adaptive LR scheme in which the LR is divided by ten every 50 epochs.



(a) NN architecture for experiment 3: it has nine inputs, 2 hidden layers with six and three nodes respectively.



(b) NN architecture for experiment 4 : it has nine inputs, two hidden layers with six and three nodes respectively.

Figure 8: Architectures of the NNs for combining with a learned fuser. The output of both the NN architectures is either: CN, MCI, or AD. A ReLu activation function was applied to the hidden layer(s) and a softmax activation function was applied to the output layer. Figures were created with alexlenail.me/NN-SVG.

In the fifth experiment, the architecture that was designed for the individual model ResNet (figure 4), was here applied as fuser. Therefore, this ensemble is referred to as *ResNetFuser1*. This experiments was done to see if more complex architectures, further improve performance over the individual models.

In the sixth experiment, the same architecture was used as in the fifth experiment. The 203 original input features were re-entered to the fuser as additional information. This experiment is referred to as *ResNetFuser2*. The input features were normalized with the min-max normalisation technique. The hypothesis is that this gives improved performance in comparison to experiment five as the network might learn other dependencies to trust or not trust certain models or data inputs from certain models.

In the final experiment, experiment 7, the same inputs were used as in experiment 6. However, in this experiment, they were split into the nine relative probabilities from the individual algorithms and the 203 features. The 203 features are entered to the first ResNet and downscaled to three features. These are then added to the nine relative probabilities from the individual models, adding up to twelve inputs for the second ResNet. It is hypothesised that this experiment results in even better performance, as the first ResNet might learn which features are resulting in improved performance from the second ResNet. Since two ResNets are used concurrently, this ensemble is referred to as *DoubleResNet*.

In experiment 5-7, optimisation of the validation accuracy was supported by the adaptive LR

'Reduce LR on plateau'. The same settings were applied as for optimisation of the individual algorithm ResNet. The optimal settings for optimal accuracy on the validation set in each of experiments five, six, and seven is a network with ten residual blocks, fifty nodes per layer, a batch size of ten, an initial LR of 5E-4, and a drop out rate of 0.05.

In all experiments with NNs, the cross-entropy loss and adam optimizer were used as these are fast and suitable for classification problems. The relu activation function was applied in all the layers, except in the output layer. The activation function applied in the output layer is the SoftMax. Hyperparameter optimisation was performed manually on the validation set.

Observe that there is increasing model complexity across experiments 4 to 7, sustaining enhanced flexibility.

2.6 Performance evaluation

Classification performance of individual models and ensemble models were evaluated with the balanced classification accuracy (BCA) and the multi-class area under the curve (mAUC). The performance evaluation framework was adopted from the TADPOLE challenge.

The mAUC represents the area under the ROC curve (AUC) applicable for problems with more than two classes Hand and Till (2001). All the class i and class j data points are ranked in increasing order based on the probability that a data point x belongs to class i. S_i is then the sum of the ranks of the data points belonging to class i. The AUC $\hat{A}(c_i|c_j)$ for classification of one class against another class (in this case c_i and c_j respectively) is given by:

$$\hat{A}(c_i|c_j) = \frac{S_i - n_i(n_i+1)/2}{n_i n_j} \tag{1}$$

with n_i and n_j are the number of data points belonging to class i and class j respectively. The average AUC is defined as:

$$\hat{A}(c_i, c_j) = \frac{1}{2}(\hat{A}(c_i|c_j) + \hat{A}(c_j|c_i)) \tag{2}$$

Then the mAUC is given by:

$$mAUC = \frac{2}{L(L-1)} \sum_{i=2}^{L} \sum_{j=1}^{i} \hat{A}(c_i, c_j) \tag{3}$$

with L the number of classes.

The BCA is derived from the classification accuracy and compensates for class imbalance in the data Brodersen et al. (2010). This classification method uses discrete data points as input. In the case of the TADPOLE challenge, the inputs are the classes of the diagnosis': CN, MCI, and AD. The the BCA for class i looks as follows:

$$BCA_i = \frac{1}{2}\left[\frac{TP}{TP+FN} + \frac{TN}{TN+FP}\right] \tag{4}$$

with TP, FP, TN, FN representing the number of true positives, false positives, true negatives, and false negatives for classification of class *i*. The total BCA, for all classes, can be calculated by taking the mean of the BCA for each class.

The variance within the test set was accounted for by performing 100 bootstraps on the test set.

The significance of the difference between models was evaluated by the non-parametric McNemar Chi-square test (Dietterich, 1998). This test is specifically applicable for pairwise comparisons, which is the case in this study. With contingency table 4, the McNemar test statistic is

$$\chi^2 = \frac{(N^{10} - N^{01})^2}{N^{10} + N^{01}} \tag{5}$$

Moreover, the diversity between individual algorithms and ensemble models was assessed by the pairwise diversity measure: the correlation coefficient ρ (Kuncheva and Whitaker, 2003). This calculates the correlation between two binary classifier outputs (true or false). With contingency table 4, the correlation coefficient ρ between two binary classifiers is

$$\rho_{1,2} = \frac{N^{11}N^{00} - N^{01}N^{10}}{\sqrt{(N^{11} + N^{10})(N^{01} + N^{00})(N^{11} + N^{01})(N^{10} + N^{00})}} \tag{6}$$

Two classifiers can be considered diverse if they make different errors on the same data points (Dietterich, 2000). Generally, it is known that differences between individual classifiers' outputs contribute towards overall ensemble accuracy. However, there is no formal proof for this in the classification context. Kuncheva et al. conclude that one should be careful with these measures since an improved accuracy is not always observed when combining diverse classifiers (Kuncheva and Whitaker, 2003). Nonetheless, we still rapport the diversity to give more insight on classifier performance and possible benefits that could be obtained by combination.

Table 4: Example contingency table. C_1 - classifier 1, C_2 - classifier 2, N^{11} - number of points correctly classified by both C_1 and C_2, N^{10} - number of points correctly classified by C_1 and wrongly classified by C_2, N^{01} - number of points wrongly classified by C_1 and correctly classified by C_2, N^{00} - number of points wrongly classified by both C_1 and C_2.

	C_1 correct	**C_1 wrong**
C_2 correct	N^{11}	N^{10}
C_2 wrong	N^{01}	N^{00}

2.7 Implementation

The code for this study was written in Python 3.7.9, using the open source neural network library Pytorch, using torch version 1.8.0. The code can be downloaded from (github.com/lottemulder/ensemble4AD).

3 Results

3.1 TADPOLE-SHARE

The collection of algorithms for predicting progression in AD resulted in eight algorithms, from which five were reused in this study. Despite the organisation of a hackathon, re-using code from other researchers was experienced to be challenging. First of all, not all participants submitted their algorithms in time. Three out of eight algorithms were submitted to the GitHub repository within the deadline of the hackathon (these were the benchmark algorithms and the algorithm EMCEB of the main organiser of the hackathon). Four other algorithms were submitted to the GitHub repository a few months after the hackathon deadline, and one has not yet been submitted. Second, not all of the algorithms were submitted according to the instructions described in the methods section. For example, one of the algorithms did not have separate training and predict functions, but instead, these algorithms have one function for training and prediction. Furthermore, another algorithm used the researcher's own split function, instead of the data split function that was provided by the hackathon organisers. Third, two of the algorithms were originally (mainly) written in R, whereas the preferred programming language for the TADPOLE-SHARE project was Python. These algorithms were rewritten so that R was called from Python. This complicated the reuse of these algorithms on different data sets. Fourth, algorithms were not always bug-free submitted to the GitHub repository. Lastly, algorithms were sometimes hard-coded for training or testing on a specific data set. To train or test the algorithm on different data sets, the algorithms had to be adjusted. Despite extensive effort to rewrite and debug algorithms allowing for retraining and retesting, only three out of eight algorithms were retrainable and retestable.

3.2 Performance evaluation

Performance of the individual algorithms and the ensemble models were evaluated and compared. The comparison of the models was split into two separate situations: the comparison of; (1) the individual models and the unlearned ensembles both trained according to the single training set (results presented in table 5); and (2) the individual algorithms, the unlearned ensemble models and the learned ensemble models all trained according to the separate training set scenario (results presented in table 6). Each individual algorithm and ensemble model is evaluated on both metrics BCA and mAUC. More insight on results is provided with boxplots.

Table 5: BCA and mAUC scores (mean [std]) of individual algorithms and unlearned ensembles in the single training set scenario obtained with 100 times bootstrapping on X_{Test}.

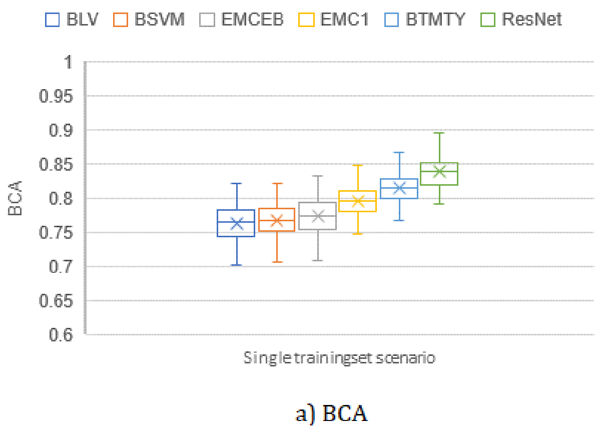
Single training set scenario					
Individual algorithms	**BCA**	**mAUC**	*Unlearned ensembles*	**BCA**	**mAUC**
BLV	0.764 [±0.027]	0.756 [±0.028]	*MeanAll*	0.805 [±0.027]	0.897 [±0.020]
BSVM	0.767 [±0.027]	0.792 [±0.022]	*MeanBest*	0.818 [±0.025]	0.910 [±0.019]
EMCEB	0.773 [±0.025]	0.884 [±0.024]	*MedianAll*	0.803 [±0.026]	0.915 [±0.018]
EMC1	0.795 [±0.023]	0.902 [±0.019]	*MedianBest*	0.821 [±0.023]	0.905 [±0.018]
BTMTY	0.812 [±0.024]	0.889 [±0.025]			
ResNet	0.838 [±0.022]	0.898 [±0.021]			

Table 6: BCA and mAUC scores (mean [std]) of individual algorithms, unlearned and learned ensembles in the separate training set scenario obtained with 100 times bootstrapping on X_{Test}. Note that EMC1, BTMTY and ResNet were not used in the both the unlearned and learned ensembles.

Separate training set scenario					
Individual algorithms	**BCA**	**mAUC**	*Unlearned ensembles*	**BCA**	**mAUC**
BLV	0.763 [±0.022]	0.756 [±0.029]	*MeanAll*	0.762 [±0.023]	0.858 [±0.022]
BSVM	0.768 [±0.021]	0.798 [±0.024]	*MeanBest*	0.769 [±0.026]	0.865 [±0.022]
EMCEB	0.761 [±0.029]	0.866 [±0.020]	*MedianAll*	0.760 [±0.028]	0.848 [±0.022]
			MedianBest	0.770 [±0.027]	0.864 [±0.019]
Learned ensembles	**BCA**	**mAUC**			
OneNeuron	0.805 [±0.025]	0.840 [±0.030]			
TwoLayers	0.774 [±0.026]	0.876 [±0.020]			
ResNetFuser1	0.795 [±0.027]	0.878 [±0.020]			
ResNetFuser2	0.809 [±0.026]	0.893 [±0.020]			
DoubleResNet	0.814 [±0.023]	0.902 [±0.020]			

3.2.1 Individual models

Figure 9 and table 5 show the prediction performance of the six individual algorithms trained according to the single training set scenario. Observe that a slight difference in scores is obtained by the TADPOLE-SHARE algorithms in table 5 compared with the scores obtained by the same algorithms in the TADPOLE challenge in table 1. This slight difference is caused by rewriting the code according to the TADPOLE-SHARE pipeline. Algorithms EMCEB, EMC1, BTMTY, and ResNet score significantly better than benchmark model BLV ($p<0.05$) (McNemar test)). ResNet obtained the highest BCA score (0.834 [±0.024]) and model EMC1 obtained the highest mAUC score (0.902 [±0.019]). Neither ResNet nor EMC1 scored significantly better than BTMTY, EMCEB, and BSVM. Observe that a difference exists between BCA and mAUC scores. This difference could be explained by the fact that the BCA takes hard classes into account while the mAUC is calculated by taking the sum of the ranks of the relative probabilities. For example, EMC1 is performing worse than BTMTY and ResNet on the BCA, while performing best on the mAUC. This might indicate that algorithm EMC1 might classify less correct, alternatively is more confident on correctly classified points than e.g. BTMTY and EMC1. In our opinion, both being correct as being confident are important properties for an algorithm. To that end, both metrics are computed. Furthermore, these results are in line with expectations as the performance increases with a more advanced methodology.



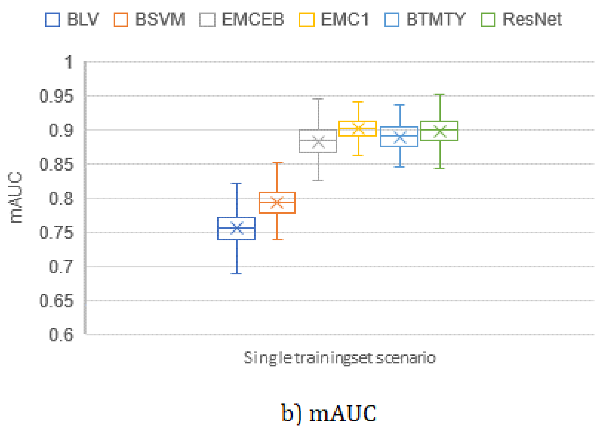


Figure 9: Boxplots of the BCA (a) and mAUC (b) scores obtained with 100x bootstrapping on X_{Test}. The plots show the six individual models: BLV, BSVM, EMCEB, EMC1, BTMTY and ResNet. Observe that all of these models were trained according to the single training set scenario. Boxplots indicate the median (the horizontal line in each box), the mean (the cross in each box), the interquantile range (IQR) of the data (the boxes that span the 25^{th} to the 75^{th} percentile) and the whiskers (the extending vertical lines), indicating the minimum (25^{th} percentile - 1.5 x IQR) and the maximum (75^{th} percentile + 1.5 x IQR).

Figure 10 provides a comparison of the individual models trained according to the single training set scenario and the separate training set scenario. Table 6 shows the prediction performance of algorithms BLV, BSVM, EMCEB, and ResNet trained according to the separate training set scenario. Concluding from figure 10, training on half of the data does not have a significant effect on models BLV and BSVM. For algorithm BLV, which takes the previous visit diagnosis as the follow-up visit diagnosis, it is evident that reducing the data does not affect the performance as the data split was stratified on converters and classes. For algorithm BSVM, similar performance for both scenarios might be explained by the fact that this model is relatively simple. Therefore, reducing the training data by half does not affect the performance. The performance of the model ResNet decreased from 0.838 to 0.785 (0.053 difference) for the BCA. Likewise, the performance of model EMCEB decreases slightly from 0.774 to 0.761 (0.013 difference) for the BCA and from 0.883 to 0.866 (0.017 difference) for mAUC when training in the separate training set scenario. However, these differences (for both BCA and mAUC scores) are within the standard deviations of either of the performances in both scenarios and not significantly different ($p>0.05$). Though, this slight decrease in performance might be explained by the fact that algorithms EMCEB and ResNet are more complex than BSVM and BLV.

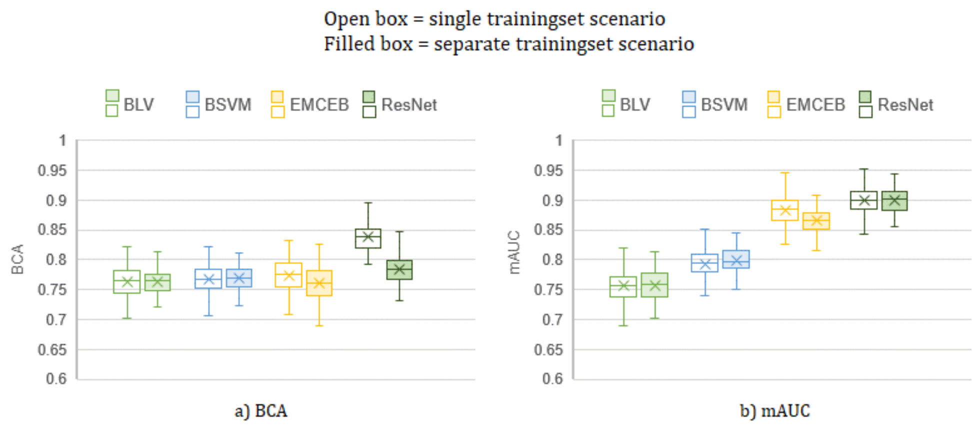
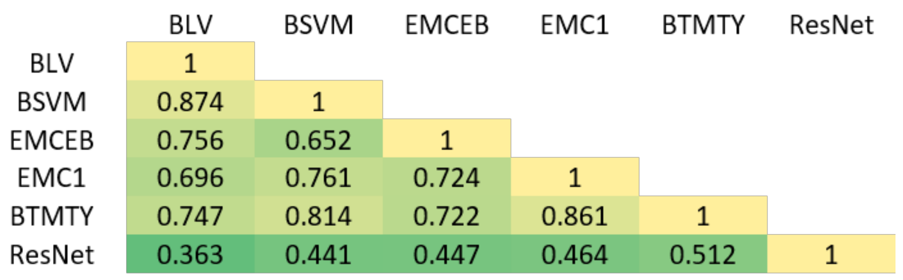


Figure 10: A comparison between individual models trained according to the single training set scenario (open boxes) and separate training set scenario (filled boxes). Boxplots of the BCA (a) and mAUC (b) scores obtained with 100x bootstrapping on X_{Test}. The plot only shows the four retrainable individual models: BLV, BSVM, EMCEB, and ResNet.

Table 7 shows the correlation matrix of the six individual algorithms trained according to the single training set scenario. Most algorithms correlate strongly ($\rho > 0.5$). ResNet shows moderate correlation with algorithms BLV, BSVM, EMCEB, and EMC1 ($\rho < 0.5$). These results indicate that combining the six algorithms with unlearned fusers would not necessarily lead to improvements in performance. Similarly, these results indicate that combining the three retrainable and retestable algorithms BLV, BSVM, EMCEB with unlearned and learned fusers, would also not necessarily lead to improvements in performance, while ResNet could be a valuable addition.

Table 7: Correlation coefficient ρ for the individual algorithms in the single training set scenario.



	BLV	BSVM	EMCEB	EMC1	BTMTY	ResNet
BLV	1					
BSVM	0.874	1				
EMCEB	0.756	0.652	1			
EMC1	0.696	0.761	0.724	1		
BTMTY	0.747	0.814	0.722	0.861	1	
ResNet	0.363	0.441	0.447	0.464	0.512	1

3.2.2 Single training set scenario

Figure 11 and table 5 presents the prediction performance of individual algorithms (a,c) and the unlearned ensembles MeanAll, MeanBest, MedianAll, MedianBest (b,d). The mean results of the ensembles can be found in table 5. None of the unlearned ensembles performed better than the best performing individual algorithm on the BCA (ResNet with BCA score of 0.838 [±0.024]). However, on the mAUC, the ensembles MeanBest, MedianAll, and MedianBest perform slightly better (MedianAll obtained a mAUC = 0.915 [±0.018]) than the best performing individual algorithm (EMC1 with BCA score of 0.902 [±0.019]). Yet, these differences are not significant (p>0.05). The slightly improved performance of the mean and median is according to our expectations as the mean and median ensembles might compensate for the under- and overestimation of individual algorithms.

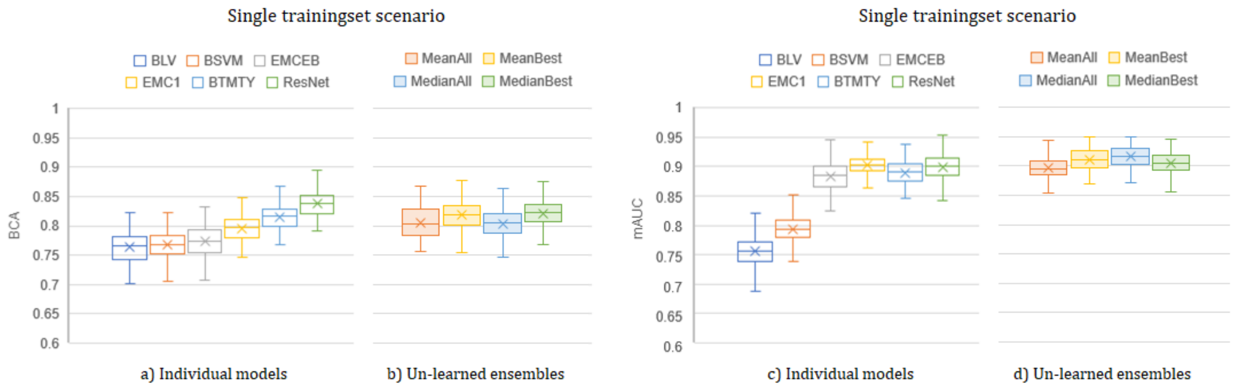


Figure 11: Boxplots of the BCA (a,b) and mAUC (c,d) scores obtained with 100 times bootstrapping on X_{Test}. The plots show all six individual models (non-filled boxplots (a,c)) and unlearned ensemble models (filled boxplots (b,d)). Observe that all of these models were trained according to the single training set scenario. MeanAll - the ensemble with the mean fuser of all six individual algorithms. MeanBest - the best performing ensemble with the mean fuser on X_{Test} (ensemble of EMC1 and ResNet). MedianAll - the ensemble with the median fuser of all six individual algorithms with the median fuser. MedianBest - the best performing ensemble with the median fuser on X_{Test} (ensemble of EMC1 and ResNet)

3.2.3 Separate training set scenario

Figure 12, 13 and table 6 present the prediction performances of individual algorithms, and unlearned and learned ensembles both fusing retrainable algorithms BLV, BSVM, EMCEB, trained in the separate training set scenario. Results of the unlearned ensembles fusing BLV, BSVM, EMCEB and ResNet are presented in Appendix A (table 8 and figure 14).

Observe that the unlearned ensembles MeanBest and MedianBest have slightly higher performance on the BCA than the best scoring individual algorithm EMCEB (p>0.05). None of the unlearned ensembles has higher performance than EMCEB on the mAUC score. The best performing unlearned ensemble is MeanBest (BCA = 0.769 [±0.026]; mAUC = 0.865 [±0.022]).

The performance increases with the increased complexity of the learned fusers in figures 12 and 13. The BCA for ensemble OneNeuron is unexpectedly high, compared to the individual algorithms. As this was the first conducted experiment with learned fusers, this high performance might be due to better optimisation in model selection compared to other learned ensembles. On the other hand, the mAUC score of OneNeuron is not improved over the mAUC of the best scoring individual algorithm used here, EMCEB. On the BCA, ensemble ResNetFuser1 is performing slightly better than ensemble TwoLayers, implying that the more complex ResNetFuser1 is better. However, on the mAUC score, these two ensembles have similar performances. Ensembles ResNetFuser2 and DoubleResNet fuse the individual algorithms and the 203 input features. It is evident from figure 12 and 13 that, besides fusing the individual algorithms, also re-entering the 203 features to the fuser, therewith giving the ensemble extra information, moderately improves performance compared to individual algorithms.

DoubleResNet is the ensemble scoring best on both BCA (0.814 [±0.023]) and mAUC (0.902 [±0.020]). Although DoubleResNet improves with 0.053 on the BCA and 0.036 on the mAUC compared with the best performing individual algorithm (EMCEB), results are not statistically significant ($p>0.05$).

Comparing the best scoring unlearned ensemble with the best scoring learned ensemble, we can conclude that the learned ensemble DoubleResNet outperforms the unlearned ensemble MeanBest, however not statistically significant ($p>0.05$).

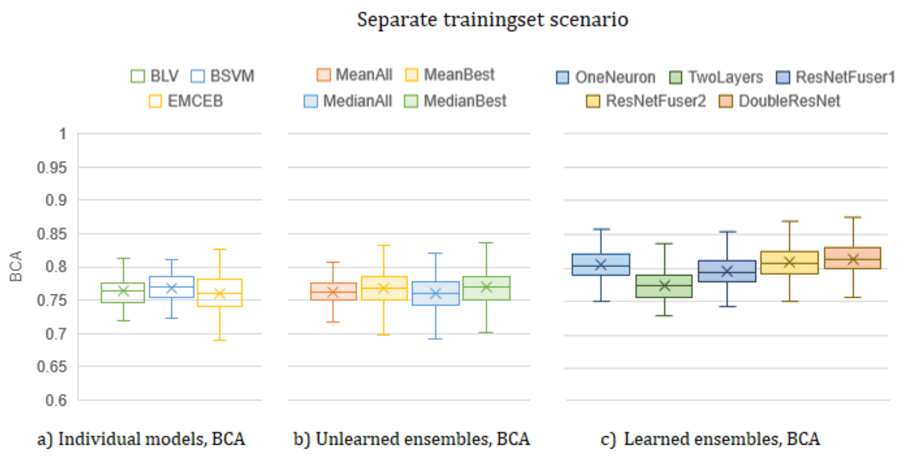


Figure 12: Boxplots of the BCA scores of individual algorithms (a), unlearned ensembles (b) and, learned ensembles (c) obtained with 100x bootstrapping on X_{Test}. Observe that all of these models were trained according to the separate training set scenario. MeanAll - the ensemble fusing the three retrainable individual algorithms with the mean, MeanBest - the best performing ensemble with the mean fuser (ensemble of EMCEB and ResNet), MedianAll - the ensemble fusing the three retrainable individual algorithms with the median fuser, MedianBest - the best performing ensemble with the median fuser (ensemble of EMCEB and ResNet)

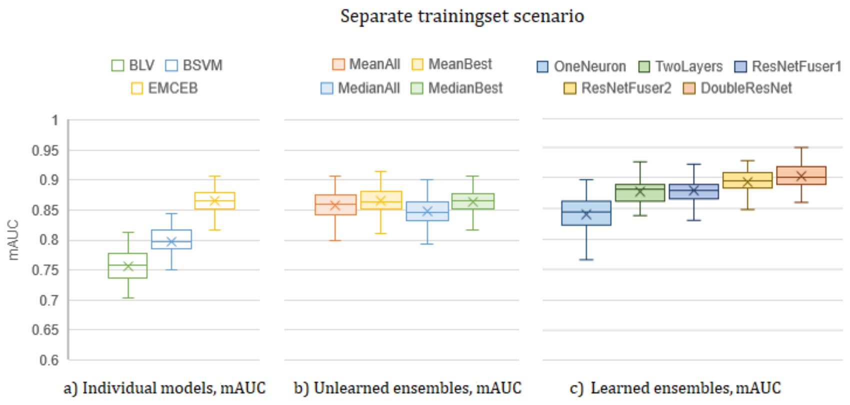


Figure 13: Boxplots of the mAUC scores of individual algorithms (a), unlearned ensembles (b) and, learned ensembles (c) obtained with 100x bootstrapping on X_{Test}. Observe that all of these models were trained according to the separate training set scenario. MeanAll - the ensemble fusing the three retrainable individual algorithms with the mean, MeanBest - the best performing ensemble with the mean fuser (ensemble of EMCEB and ResNet), MedianAll - the ensemble fusing the three retrainable individual algorithms with the median fuser, MedianBest - the best performing ensemble with the median fuser (ensemble of EMCEB and ResNet)

4 Discussion

In this study, we collected existing optimised algorithms predicting the progression of AD. We investigated the possibilities of how to combine these collected, *strong* algorithms. We presented two frameworks for careful combination with unlearned and learned fusers. Finally, we evaluated and compared individual algorithms and several ensembles with unlearned and learned fusers. In this section, I will discuss the results and the limitations of this study and I will propose ideas for future work.

4.1 TADPOLE-SHARE

The first aim of this study was to collect existing, optimised algorithms for the predicting progression in AD. To this end, we offered TADPOLE-SHARE to allow for sharing scientific code among the research community. Participants were asked to rewrite their code in a particular format that allowed for re-training and testing on different data sets. However, sharing and reusing code proved challenging. Here, I will summarise the main obstacles that delayed or precluded the combination study.

Most of the collected algorithms were not immediately retrainable and/or retestable on different data sets. The main reason for this was hard-coding towards specific data of the train and/or predict function. As retraining and retesting algorithms are necessary for combination with learned fusers, algorithms EMC1, BTMTY were excluded from the study with learned fusers. Furthermore, some of the algorithms used another programming language (R) within their final python code. This complicated running and debugging their code.

For future projects that aim to share code with the scientific community, organisers should be strict on participants and should have a test available that checks if the algorithm is in the right format and should reject submissions that are not. This test could for example consist of running the algorithm on a test data set consisting of one patient from an external data set (that is not included in the current test data set). Moreover, it might help motivate participants by giving them insight into the benefits of their hard work rewriting the code. Being clear on the kind of follow-up project their algorithm might participate in after sharing, might motivate the participants prioritising to submit code in the right format.

4.2 Individual model ResNet

The individual model ResNet was designed for this study because only five algorithms from TADPOLE-SHARE could be combined with unlearned fuser methods and, only three could be combined with learned fuser methods. ResNet performed best on the BCA score (0.838 [$\pm$0.022]) and third on the mAUC score (0.898 [$\pm$0.021]) in comparison to the other individual algorithms deployed in this study. It would have been ranked 8th based on the mAUC performance in the TADPOLE challenge. The complex residual network in combination with the smart *'reduce LR on plateau'* adaptive LR scheduler, might have caused its relatively good performance. Although ResNet is performing relatively well already, more extensive optimisation of this model might lead to further improvement of the performance.

ResNet was, thus far, only tested on X_{Test}. This allowed using this algorithm for fusing with unlearned methods. However, for this algorithm to be used for fusing with learned ensembles, predictions should be made on X_{Train}^{Ind} so that these predictions can be entered as training data to the fuser (see figure 7). From the correlation matrix in table 7, it is clear that ResNet correlates the least to any of the other algorithms. Therefore, adding this algorithm to learned ensembles might improve the ensemble's performance.

4.3 Ensemble results

The second aim of this study was to investigate the possibility of combining *strong* optimised algorithms for predicting progression in AD by providing frameworks for combining with unlearned and learned fusers. The third aim was to evaluate and compare individual algorithms and several ensembles with unlearned and learned fusers.

Although improvements were not significant, reusing optimised algorithms and combining them led to further improvement of algorithms' performance. Ensembles with learned methods outperformed ensembles with unlearned methods on both the BCA and the mAUC. Gradual improvement was obtained by deploying more complex fusers. The largest improvement was obtained by the learned ensemble DoubleResNet, the most complex learned fuser experimented with. This ensemble fused the retrainable algorithms BLV, BSVM, EMCEB, and reused the original data features as additional feed-in. Fusion methodology consists of two dependent ResNets. DoubleResNet scored 0.814 [$\pm$0.023] on the BCA and 0.902 [$\pm$0.020] on the mAUC while the best algorithm among the fused algorithms (EMCEB) scores 0.761 [$\pm$0.029] on the BCA and 0.866 [$\pm$0.020] on the mAUC.

The overall best performing model on the BCA is the individual algorithm ResNet trained according to the single training set scenario (BCA=0.838 [$\pm$0.022]). None of the ensemble methods that, amongst others, fuses individual algorithm ResNet, improves on the BCA. The overall best performing model on the mAUC is unlearned ensemble MedianAll trained according to the single training set scenario (mAUC= 0.915 [$\pm$0.018]). This best performing ensemble on the mAUC fuses all six individual algorithms and is therefore not directly comparable with DoubleResNet.

Learned ensembles ResNetFuser1, ResNetFuser2 and DoubleResNet use the same architecture as individual algorithm ResNet. However, as the three enumerated learned ensembles do not fuse ResNet, we can not directly compare them with individual algorithm ResNet. Nevertheless, we could argue, if it is worth the effort to combine algorithms or rather develop and further optimise our own method. ResNet (trained according to the single training set scenario) outperforms DoubleResNet on the BCA and, mAUC scores are similar. On the other hand, the ensembles with learned fusers, were not carefully optimised. Doing so might result in further improvements.

Thus far, learned ensembles only fused strongly correlating algorithms ($\rho > 0.5$) BLV, BSVM, and EMCEB. Future research should point out how fusing ResNet affects the performance of learned ensembles, especially since it correlates moderately with other individual algorithms.

4.4 Limitations

This study has several limitations. The data was split in X_{Train} and X_{Test}, dividing the data into time points before January 1, 2018, and time points after January 1, 2018. This split was made originally in the TADPOLE challenge to mimic clinical settings. Algorithms learn from data of previous time points of subjects and forecast biomarkers such as clinical diagnosis for time points in the future. However, from a machine learning point of view, this limits this study as X_{Test} is not independent of X_{Train}. However, some improvement could be made to make $Split_1$ a better choice. Future research could incorporate forecasts for visits farther in the future from which the truth exists in the data set. Currently, the individual algorithms' forecast that was used to train the future, was limited to only the follow-up visit in the data set. The follow-up time difference in data set X_{Train} is, on average, 2 years shorter than in X_{Test} (follow-up time in X_{Train} is 0.6 years and in X_{Test} is 2.6 years). Forecasting on multiple future visits could decrease the difference of follow-up time in X_{Train} and X_{Test}.

Algorithm BLV takes the last visit diagnosis as the diagnosis for the next visit. Although this algorithm does not learn anything, it obtained a BCA of 0.764 [$\pm$0.027] and an mAUC score of 0.756 [$\pm$0.028]. The data consists of subjects having a stable diagnosis throughout the whole ADNI study, and subjects that convert to a different diagnosis. As data sets X_{Train} and X_{Test} consist only of 27.1% and 16.7% converting participants respectively, an algorithm such as BLV can seem to have a quite high performance. In this study, the BCA and mAUC are not calculated separately for converters and non-converters. Therefore, it is unknown if the slight improvements on these two scores for the ensembles are through slightly improved performance on converters and similar performance on non-converters, or vice versa. More insight into the strengths and weaknesses of algorithms could be obtained by measuring the performance of converters and non-converters separately.

Another limitation of this study is that only three, quite strongly correlating, algorithms were combined. In fact, two out of three algorithms are benchmark models which were designed by the organisers of the TADPOLE challenge and served as an example and comparison for participants. Therefore, we can not consider these algorithms as *strong*, as *strong* algorithms were defined as independently optimised algorithms. Unfortunately, the strongest algorithms collected with TADPOLE-SHARE (EMC1 and BTMTY) were not submitted in a way that allowed for retraining, despite extensive efforts to facilitate this. As such, these algorithms could not be used in experiments with learned fusers. Nevertheless, the preliminary results of this study show that combining *strong* algorithms in principle, slightly improved performance.

Finally, the use of the data from ADNI brings some limitations. First of all, the ADNI study included healthy volunteers and volunteers that visited the clinic with memory complaints. As memory complaints do not start to develop concurrently with measurable brain changes, but at a later stage, the ADNI database misses out on individuals within the first stretch of the AD continuum. Secondly, the ADNI data used in this study only consists of individuals with clinical diagnosis CN, MCI, and AD. However, it is unknown how the computer models will perform on data comprising all existing brain diseases. Future studies should test these models on databases such as the Rotterdam Study Ikram et al. (2017) or the UK Biobank, which do contain volunteers that are in the pre-clinical stage of AD. Furthermore, future work should assess the ability of our proposed ensemble methods to distinguish between different brain diseases.

4.5 Future work

Open science, in particular, sharing reproducible code is advancing at a rapid pace. The future holds many opportunities concerning reusing code from others. Initiatives such as the TADPOLE-SHARE project and algorithm sharing tools like grand-challenge.org provide a base for comparison, further development and refinement of existing algorithms. As there is plenty of room for improvement in this study, we propose the following ideas for future work aside from the ideas that were already proposed in the previous section.

First of all, it is largely unknown how algorithms would perform on external data. Currently, algorithms predicting progression in AD are not yet used in clinical practice. Yet, the amount of publicly available data sources on brain diseases is increasing. Studies such as ADNI (adni.loni.usc.edu), Parelsnoer (Aalten et al., 2014), the UK Biobank (ukbiobank.ac.uk) and the Rotterdam Study Ikram et al. (2017), will favor algorithm development and allow for evaluation of an algorithm's ability to generalize. We hypothesize that our learned ensembles will generalise better towards external data in comparison with the individual algorithms. Therefore, future works should focus on the generalisability towards external data of algorithms and ensembles.

Secondly, in this study, we showed preliminary results of improvement of performance when combining only three strongly correlation algorithms. Results might further improve and might become significant when more and other, less strongly correlating algorithms are combined with methods proposed in this study.

Third, we investigated neural networks as learned fusers. Instead, it might be interesting to investigate other methods would perform as fuser, such as recurrent neural network (RNN), SVMs or linear regression.

Finally, as mentioned before, from a machine learning perspective, the data split was not optimal. It could give more insight into how well ensembles perform by also optimising and evaluating them with a different data split. A future experiment could be conducted where data set X_{Train} and X_{Test} are joined and randomly split so that X_{Train} and X_{Test} do not contain overlapping subjects.

5 Conclusion

This study has given more insight into how open science can improve the performance of algorithms predicting the progression in AD. We show that reusing state-of-the-art algorithms from various institutes is challenging. Nevertheless, the TADPOLE-SHARE project and algorithm sharing initiatives like grand-challenge.org provide a base for comparison, further development and refinement of existing algorithms.

We found that combining existing algorithms slightly increases overall prediction performance, although not statistically significant. Both BCA and mAUC showed a clear trend of improved performance with increasing fuser training complexity. First of all, learned fusers outperformed unlearned fusers. Secondly, reusing the original features as additional feed-in enhanced the end performance even further. Since the results in this study were obtained by combining only three algorithms that are strongly correlated, increased performance can be expected when combining other algorithms that are less strongly correlating.

However, it is currently unknown when algorithm performance is sufficient for clinical translation. Generalisability of algorithms towards different data sets is often not examined. Nevertheless, we believe that our method will be more robust, as it combines the opinion of multiple experts. Therefore, future work should focus on algorithms' generalisability, and on on the interoperability of already existing and newly developed algorithms.

References

Pauline Aalten, Inez H.G.B. Ramakers, Geert Jan Biessels, Pete Paul de Deyn, Huiberdina L. Koek, Marcel G.M. OldeRikkert, Ania M. Oleksik, Edo Richard, Lieke L. Smits, John C. van Swieten, Laura K. Teune, Aad van der Lugt, Frederik Barkhof, Charlotte E. Teunissen, Nico Rozendaal, Frans R.J. Verhey, and Wiesje M. van der Flier. The Dutch Parelsnoer Institute - Neurodegenerative diseases; methods, design and baseline results. *BMC Neurology*, 14(1):254, dec 2014. ISSN 14712377. doi: 10.1186/s12883-014-0254-4.

Manon Ansart, Stéphane Epelbaum, Giulia Bassignana, Alexandre Bône, Simona Bottani, Tiziana Cattai, Raphael Couronne, Johann Faouzi, Igor Koval, Maxime Louis, Raphaël Couronné, Elina Thibeau-Sutre, Junhao Wen, Adam Wild, Ninon Burgos, Didier Dormont, Olivier Colliot, and Stanley Durrleman. Predicting the Progression of Mild Cognitive Impairment Using Machine Learning: A Systematic and Quantitative Review Predicting the Progression of Mild Cognitive Impairment Using Machine Learning: A Systematic and Quantitative Review. *Hal-02337815*, page 101848, oct 2021. ISSN 13618415. doi: 10.1016/j.media.2020.101848.

Jesús Avila, José J. Lucas, Mar Pérez, and Félix Hernández. Role of Tau Protein in Both Physiological and Pathological Conditions, apr 2004. ISSN 00319333.

Péter Bándi, Oscar Geessink, Quirine Manson, Marcory Van Dijk, Maschenka Balkenhol, Meyke Hermsen, Babak Ehteshami Bejnordi, Byungjae Lee, Kyunghyun Paeng, Aoxiao Zhong, Quanzheng Li, Farhad Ghazvinian Zanjani, Svitlana Zinger, Keisuke Fukuta, Daisuke Komura, Vlado Ovtcharov, Shenghua Cheng, Shaoqun Zeng, Jeppe Thagaard, Anders B. Dahl, Huangjing Lin, Hao Chen, Ludwig Jacobsson, Martin Hedlund, Melih Çetin, Eren Halici, Hunter Jackson, Richard Chen, Fabian Both, Jörg Franke, Heidi Kusters-Vandevelde, Willem Vreuls, Peter Bult, Bram Van Ginneken, Jeroen Van Der Laak, and Geert Litjens. From Detection of Individual Metastases to Classification of Lymph Node Status at the Patient Level: The CAMELYON17 Challenge. *IEEE Transactions on Medical Imaging*, 38(2):550–560, feb 2019. ISSN 1558254X. doi: 10.1109/TMI.2018.2867350.

Leo Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, aug 1996. ISSN 08856125. doi: 10.1023/A:1018054314350.

Kay H. Brodersen, Cheng Soon Ong, Klaas E. Stephan, and Joachim M. Buhmann. The balanced accuracy and its posterior distribution. In *Proceedings - International Conference on Pattern Recognition*, pages 3121–3124, 2010. ISBN 9780769541099. doi: 10.1109/ICPR.2010.764.

Esther E. Bron, Marion Smits, Wiesje M. van der Flier, Hugo Vrenken, Frederik Barkhof, Philip Scheltens, Janne M. Papma, Rebecca M.E. Steketee, Carolina Méndez Orellana, Rozanna Meijboom, Madalena Pinto, Joana R. Meireles, Carolina Garrett, António J. Bastos-Leite, Ahmed Abdulkadir, Olaf Ronneberger, Nicola Amoroso, Roberto Bellotti, David Cárdenas-Peña, Andrés M. Álvarez-Meza, Chester V. Dolph, Khan M. Iftekharuddin, Simon F. Eskildsen, Pierrick Coupé, Vladimir S. Fonov, Katja Franke, Christian Gaser, Christian Ledig, Ricardo Guerrero, Tong Tong, Katherine R. Gray, Elaheh Moradi, Jussi Tohka, Alexandre Routier, Stanley Durrleman, Alessia Sarica, Giuseppe Di Fatta, Francesco Sensi, Andrea Chincarini, Garry M. Smith, Zhivko V. Stoyanov, Lauge Sørensen, Mads Nielsen, Sabina Tangaro, Paolo Inglese, Christian Wachinger, Martin Reuter, John C. van Swieten, Wiro J. Niessen, and Stefan Klein. Standardized evaluation of algorithms for computer-aided diagnosis of dementia based on structural MRI: The CADDementia challenge. *NeuroImage*, 111:562–579, may 2015. ISSN 10959572. doi: 10.1016/j.neuroimage.2015.01.048.

Esther E Bron, Stefan Klein, Janne M Papma, Lize C Jiskoot, Vikram Venkatraghavan, Jara Linders, Pauline Aalten, Peter Paul de Deyn, Geert Jan Biessels, Jurgen A.H.R. Claassen, Huub AM Middelkoop, Marion Smits, Wiro J Niessen, John C van Swieten, Wiesje M. van der Flier, Inez H.G.B. Ramakers, and Aad van der Lugt. Cross-cohort generalizability of deep and conventional machine learning for MRI-based diagnosis and prediction of Alzheimer's disease, 2020. ISSN 23318422.

Danni Cheng, Manhua Liu, Jianliang Fu, and Yaping Wang. Classification of MR brain images by combination of multi-CNNs for AD diagnosis. In Charles M. Falco and Xudong Jiang, editors, *Ninth International Conference on Digital Image Processing (ICDIP 2017)*, volume 10420, page 1042042. SPIE, jul 2017. ISBN 9781510613041. doi: 10.1117/12.2281808.

Noel C.F. Codella, David Gutman, M. Emre Celebi, Brian Helba, Michael A. Marchetti, Stephen W. Dusza, Aadi Kalloo, Konstantinos Liopyris, Nabin Mishra, Harald Kittler, and Allan Halpern. Skin lesion analysis toward melanoma detection: A challenge at the 2017 International symposium on biomedical imaging (ISBI), hosted by the international skin imaging collaboration (ISIC). In *Proceedings - International Symposium on Biomedical Imaging*, volume 2018-April, pages 168–172. IEEE Computer Society, may 2018. ISBN 9781538636367. doi: 10.1109/ISBI.2018.8363547.

Zhengjia Dai, Chaogan Yan, Zhiqun Wang, Jinhui Wang, Mingrui Xia, Kuncheng Li, and Yong He. Discriminative analysis of early Alzheimer's disease using multi-modal imaging and multi-level characterization with multi-classifier (M3). *NeuroImage*, 59(3):2187–2195, feb 2012. ISSN 10538119. doi: 10.1016/j.neuroimage.2011.10.003.

Thomas G. Dietterich. Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms. *Neural Computation*, 10(7):1895–1923, oct 1998. ISSN 08997667. doi: 10.1162/089976698300017197.

Thomas G. Dietterich. Ensemble methods in machine learning. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 1857 LNCS, pages 1–15. Springer Verlag, 2000. ISBN 3540677046. doi: 10.1007/3-540-45014-9_1.

Yoav Freund and Robert E. Schapire. A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. *Journal of Computer and System Sciences*, 55(1):119–139, aug 1997. ISSN 00220000. doi: 10.1006/jcss.1997.1504.

Giovanni B. Frisoni, Nick C. Fox, Clifford R. Jack, Philip Scheltens, and Paul M. Thompson. The clinical use of structural MRI in Alzheimer disease, 2010. ISSN 17594758.

Joseph Gaugler, Bryan James, Tricia Johnson, Allison Marin, and Jennifer Weuve. 2019 Alzheimer's disease facts and figures. *Alzheimer's & Dementia*, 15(3):321–387, mar 2019. doi: 10.1016/j.jalz.2019.01.010.

Brian A. Gordon, Tyler M. Blazey, Yi Su, Amrita Hari-Raj, Aylin Dincer, Shaney Flores, Jon Christensen, Eric McDade, Guoqiao Wang, Chengjie Xiong, Nigel J. Cairns, Jason Hassenstab, Daniel S. Marcus, Anne M. Fagan, Clifford R. Jack, Russ C. Hornbeck, Katrina L. Paumier, Beau M. Ances, Sarah B. Berman, Adam M. Brickman, David M. Cash, Jasmeer P. Chhatwal, Stephen Correia, Stefan Förster, Nick C. Fox, Neill R. Graff-Radford, Christian la Fougère, Johannes Levin, Colin L. Masters, Martin N. Rossor, Stephen Salloway, Andrew J. Saykin, Peter R. Schofield, Paul M. Thompson, Michael M. Weiner, David M. Holtzman, Marcus E. Raichle, John C. Morris, Randall J. Bateman, and Tammie L.S. Benzinger. Spatial patterns of neuroimaging biomarker change in individuals from families with autosomal dominant Alzheimer's disease: a longitudinal study. *The Lancet Neurology*, 17(3):241–250, mar 2018. ISSN 14744465. doi: 10.1016/S1474-4422(18)30028-0.

David J. Hand and Robert J. Till. A Simple Generalisation of the Area Under the ROC Curve for Multiple Class Classification Problems. *Machine Learning*, 45(2):171–186, 2001. ISSN 08856125. doi: 10.1023/A:1010920819831.

Balazs Harangi. Skin lesion classification with ensembles of deep convolutional neural networks. *Journal of Biomedical Informatics*, 86:25–32, oct 2018. ISSN 15320464. doi: 10.1016/j.jbi.2018.08.006.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2016-Decem, pages 770–778, 2016. ISBN 9781467388504. doi: 10.1109/CVPR.2016.90.

Tobias Heimann, Brain Van Ginneken, Martin A. Styner, Yulia Arzhaeva, Volker Aurich, Christian Bauer, Andreas Beck, Christoph Becker, Reinhard Beichel, György Bekes, Fernando Bello, Gerd Binnig, Horst Bischof, Alexander Bornik, Peter M.M. Cashman, Ying Chi, Andrés Córdova, Benoit M. Dawant, Márta Fidrich, Jacob D. Furst, Daisuke Furukawa, Lars Grenacher, Joachim Hornegger, Dagmar Kainmüller, Richard I. Kitney, Hidefumi Kobatake, Hans Lamecker, Thomas Lange, Jeongjin Lee, Brian Lennon, Rui Li, Senhu Li, Hans Peter Meinzer, Gábor Németh, Daniela S. Raicu, Anne Mareike Rau, Eva M. Van Rikxoort, Mikaël Rousson, Lászlo Ruskó, Kinda A. Saddi, Günter Schmidt, Dieter Seghers, Akinobu Shimizu, Pieter Slagmolen, Erich Sorantin, Grzegorz Soza, Ruchaneewan Susomboon, Jonathan M. Waite, Andreas Wimmer, and Ivo Wolf. Comparison and evaluation of methods for liver segmentation from CT datasets. *IEEE Transactions on Medical Imaging*, 28(8):1251–1265, aug 2009. ISSN 02780062. doi: 10.1109/TMI.2009.2013851.

Chris Hinrichs, Vikas Singh, Lopamudra Mukherjee, Guofan Xu, Moo K. Chung, and Sterling C. Johnson. Spatially augmented LPboosting for AD classification with evaluations on the ADNI dataset. *NeuroImage*, 48(1):138–149, oct 2009. ISSN 10538119. doi: 10.1016/j.neuroimage.2009.05.056.

Matthew Hutson. Artificial intelligence faces reproducibility crisis Unpublished code and sensitivity to training conditions make many claims hard to verify, 2018. ISSN 10959203.

M. Arfan Ikram, Guy G.O. Brusselle, Sarwa Darwish Murad, Cornelia M. van Duijn, Oscar H. Franco, André Goedegebure, Caroline C.W. Klaver, Tamar E.C. Nijsten, Robin P. Peeters, Bruno H. Stricker, Henning Tiemeier, André G. Uitterlinden, Meike W. Vernooij, and Albert Hofman. The Rotterdam Study: 2018 update on objectives, design and main results. *European Journal of Epidemiology*, 32(9):807–850, sep 2017. ISSN 15737284. doi: 10.1007/s10654-017-0321-4.

Clifford R. Jack, David A. Bennett, Kaj Blennow, Maria C. Carrillo, Billy Dunn, Samantha Budd Haeberlein, David M. Holtzman, William Jagust, Frank Jessen, Jason Karlawish, Enchi Liu, Jose Luis Molinuevo, Thomas Montine, Creighton Phelps, Katherine P. Rankin, Christopher C. Rowe, Philip Scheltens, Eric Siemers, Heather M. Snyder, Reisa Sperling, Cerise Elliott, Eliezer Masliah, Laurie Ryan, and Nina Silverberg. NIA-AA Research Framework: Toward a biological definition of Alzheimer's disease, apr 2018. ISSN 15525279.

Josef Kittler, Mohamad Hatef, Robert P.W. Duin, and Jiri Matas. On combining classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(3):226–239, 1998. ISSN 01628828. doi: 10.1109/34.667881.

Paweł Ksieniewicz, Bartosz Krawczyk, and Michał Woźniak. Ensemble of Extreme Learning Machines with trained classifier combination and statistical features for hyperspectral data. *Neurocomputing*, 271:28–37, jan 2018. ISSN 18728286. doi: 10.1016/j.neucom.2016.04.076.

Ludmila I. Kuncheva. *Combining Pattern Classifiers: Methods and Algorithms: Second Edition*, volume 9781118315. John Wiley & Sons, Inc., Hoboken, NJ, USA, sep 2014. ISBN 9781118914564. doi: 10.1002/9781118914564.

Ludmila I. Kuncheva and Christopher J. Whitaker. Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine Learning*, 51(2):181–207, may 2003. ISSN 08856125. doi: 10.1023/A:1022859003006.

Manhua Liu, Daoqiang Zhang, and Dinggang Shen. Ensemble sparse classification of Alzheimer's disease. *NeuroImage*, 60(2):1106–1116, apr 2012. ISSN 10538119. doi: 10.1016/j.neuroimage.2012.01.055.

Razvan V. Marinescu, Neil P. Oxtoby, Alexandra L. Young, Esther E. Bron, Arthur W. Toga, Michael W. Weiner, Frederik Barkhof, Nick C. Fox, Stefan Klein, Daniel C. Alexander, and the EuroPOND Consortium. TADPOLE Challenge: Prediction of Longitudinal Evolution in Alzheimer's Disease. may 2018.

Razvan V. Marinescu, Neil P. Oxtoby, Alexandra L. Young, Esther E. Bron, Arthur W. Toga, Michael W. Weiner, Frederik Barkhof, Nick C. Fox, Arman Eshaghi, Tina Toni, Marcin Salaterski, Veronika Lunina, Manon Ansart, Stanley Durrleman, Pascal Lu, Samuel Iddi, Dan Li, Wesley K. Thompson, Michael C. Donohue, Aviv Nahon, Yarden Levy, Dan Halbersberg, Mariya Cohen, Huiling Liao, Tengfei Li, Kaixian Yu, Hongtu Zhu, José G. Tamez-Peña, Aya Ismail, Timothy Wood, Hector Corrada Bravo, Minh Nguyen, Nanbo Sun, Jiashi Feng, B. T. Thomas Yeo, Gang Chen, Ke Qi, Shiyang Chen, Deqiang Qiu, Ionut Buciuman, Alex Kelner, Raluca Pop, Denisa Rimocea, Mostafa M. Ghazi, Mads Nielsen, Sebastien Ourselin, Lauge Sørensen, Vikram Venkatraghavan, Keli Liu, Christina Rabe, Paul Manser, Steven M. Hill, James Howlett, Zhiyue Huang, Steven Kiddle, Sach Mukherjee, Anaïs Rouanet, Bernd Taschler, Brian D.M. Tom, Simon R. White, Noel Faux, Suman Sedai, Javier de Velasco Oriol, Edgar E.V. Clemente, Karol Estrada, Leon Aksman, Andre Altmann, Cynthia M. Stonnington, Yalin Wang, Jianfeng Wu, Vivek Devadas, Clementine Fourrier, Lars Lau Raket, Aristeidis Sotiras, Guray Erus, Jimit Doshi, Christos Davatzikos, Jacob Vogel, Andrew Doyle, Angela Tam, Alex Diaz-Papkovich, Emmanuel Jammeh, Igor Koval, Paul Moore, Terry J. Lyons, John Gallacher, Jussi Tohka, Robert Ciszek, Bruno Jedynak, Kruti Pandya, Murat Bilgel, William Engels, Joseph Cole, Polina Golland, Stefan Klein, and Daniel C. Alexander. The Alzheimer's disease prediction of longitudinal evolution (tadpole) challenge: Results after 1 year follow-up, feb 2020. ISSN 23318422.

Dev Mehta, Robert Jackson, Gaurav Paul, Jiong Shi, and Marwan Sabbagh. Why do trials for Alzheimer's disease drugs keep failing? A discontinued drug perspective for 2010-2015. *Expert Opinion on Investigational Drugs*, 26(6):735–739, jun 2017. ISSN 17447658. doi: 10.1080/13543784.2017.1323868.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

Alzheimer's Disease International Prince M, Wimo A, Guerchet M. World Alzheimer Report 2015:

The Global Impact of Dementia — Alzheimer's Disease International. *World Alzheimer Report 2013*, pages 1–87, 2015.

Robert E Schapire. The Strength of Weak Learnability. *Machine Learning*, 5(2):197–227, 1990. ISSN 15730565. doi: 10.1023/A:1022648800760.

Arnaud Arindra Adiyoso Setio, Alberto Traverso, Thomas de Bel, Moira S.N. Berens, Cas van den Bogaard, Piergiorgio Cerello, Hao Chen, Qi Dou, Maria Evelina Fantacci, Bram Geurts, Robbert van der Gugten, Pheng Ann Heng, Bart Jansen, Michael M.J. de Kaste, Valentin Kotov, Jack Yu Hung Lin, Jeroen T.M.C. Manders, Alexander Sóñora-Mengana, Juan Carlos García-Naranjo, Evgenia Papavasileiou, Mathias Prokop, Marco Saletta, Cornelia M. Schaefer-Prokop, Ernst T. Scholten, Luuk Scholten, Miranda M. Snoeren, Ernesto Lopez Torres, Jef Vandemeulebroucke, Nicole Walasek, Guido C.A. Zuidhof, Bram van Ginneken, and Colin Jacobs. Validation, comparison, and combination of algorithms for automatic detection of pulmonary nodules in computed tomography images: The LUNA16 challenge. *Medical Image Analysis*, 42:1–13, dec 2017. ISSN 13618423. doi: 10.1016/j.media.2017.06.015.

Bram van Ginneken, Samuel G. Armato, Bartjan de Hoop, Saskia van Amelsvoort-van de Vorst, Thomas Duindam, Meindert Niemeijer, Keelin Murphy, Arnold Schilham, Alessandra Retico, Maria Evelina Fantacci, Niccol Camarlinghi, Francesco Bagagli, Ilaria Gori, Takeshi Hara, Hiroshi Fujita, Gianfranco Gargano, Roberto Bellotti, Sabina Tangaro, Lourdes Bolaos, Francesco De Carlo, Piergiorgio Cerello, Sorin Cristian Cheran, Ernesto Lopez Torres, and Mathias Prokop. Comparing and combining algorithms for computer-aided detection of pulmonary nodules in computed tomography scans: The ANODE09 study. *Medical Image Analysis*, 14(6):707–722, dec 2010. ISSN 13618415. doi: 10.1016/j.media.2010.05.005.

D. Van Vliet, M. E. De Vugt, C. Bakker, Y. A.L. Pijnenburg, M. J.F.J. Vernooij-Dassen, R. T.C.M. Koopmans, and F. R.J. Verhey. Time to diagnosis in young-onset dementia as compared with late-onset dementia. *Psychological Medicine*, 43(2):423–432, feb 2013. ISSN 00332917. doi: 10.1017/S0033291712001122.

Dallas P. Veitch, Michael W. Weiner, Paul S. Aisen, Laurel A. Beckett, Nigel J. Cairns, Robert C. Green, Danielle Harvey, Clifford R. Jack, William Jagust, John C. Morris, Ronald C. Petersen, Andrew J. Saykin, Leslie M. Shaw, Arthur W. Toga, and John Q. Trojanowski. Understanding disease progression and improving Alzheimer's disease clinical trials: Recent highlights from the Alzheimer's Disease Neuroimaging Initiative, jan 2019. ISSN 15525279.

Vikram Venkatraghavan, Esther E. Bron, Wiro J. Niessen, and Stefan Klein. Disease progression timeline estimation for Alzheimer's disease using discriminative event based modeling. *NeuroImage*, 186:518–532, feb 2019. ISSN 10959572. doi: 10.1016/j.neuroimage.2018.11.024.

Lisa Vermunt, Sietske A.M. Sikkes, Ardo van den Hout, Ron Handels, Isabelle Bos, Wiesje M. van der Flier, Silke Kern, Pierre Jean Ousset, Paul Maruff, Ingmar Skoog, Frans R.J. Verhey, Yvonne Freund-Levi, Magda Tsolaki, Åsa K. Wallin, Marcel Olde Rikkert, Hilkka Soininen, Luisa Spiru, Henrik Zetterberg, Kaj Blennow, Philip Scheltens, Graciela Muniz-Terrera, Pieter Jelle Visser, B. Vellas, E. Reynish, P. J. Ousset, S. Andrieu, A. Burns, F. Pasquier, G. Frisoni, E. Salmon, J. P. Michel, D. S. Zekry, M. Boada, J. F. Dartigues, M. G.M. Olde-Rikkert, A. S. Rigaud, B. Winblad, A. Malick, A. Sinclair, L. Frölich, Philip Scheltens, C. Ribera, J. Touchon, P. Robert, A. Salva, G. Waldemar, R. Bullock, Magda Tsolaki, G. Rodriguez, Luisa Spiru, R. W. Jones, G. Stiens, G. Stoppe, M. Eriksdotter Jönhagen, A. Cherubini, P. M. Lage, T. Gomez-Isla, V. Camus, E. Agüera-Morales, F. Lopez, S. Savy, C. Cantet, and N. Coley. Duration of preclinical, prodromal, and dementia stages of Alzheimer's disease in relation to age, sex, and APOE genotype. *Alzheimer's and Dementia*, 15(7):888–898, jul 2019. ISSN 15525279. doi: 10.1016/j.jalz.2019.04.001.

Alex Ward, Sarah Tardiff, Catherine Dye, and H. Michael Arrighi. Rate of Conversion from Prodromal Alzheimer's Disease to Alzheimer's Dementia: A Systematic Review of the Literature. *Dementia and Geriatric Cognitive Disorders Extra*, 3(1):320–332, sep 2013. ISSN 1664-5464. doi: 10.1159/000354370.

Michael W. Weiner, Dallas P. Veitch, Paul S. Aisen, Laurel A. Beckett, Nigel J. Cairns, Robert C. Green, Danielle Harvey, Clifford R. Jack, William Jagust, John C. Morris, Ronald C. Petersen, Andrew J. Saykin, Leslie M. Shaw, Arthur W. Toga, and John Q. Trojanowski. Recent publications from the Alzheimer's Disease Neuroimaging Initiative: Reviewing progress toward improved AD clinical trials, apr 2017. ISSN 15525279.

Mark D. Wilkinson, Michel Dumontier, IJsbrand Jan Aalbersberg, Gabrielle Appleton, Myles Axton, Arie Baak, Niklas Blomberg, Jan Willem Boiten, Luiz Bonino da Silva Santos, Philip E. Bourne, Jildau Bouwman, Anthony J. Brookes, Tim Clark, Mercè Crosas, Ingrid Dillo, Olivier Dumon, Scott Edmunds, Chris T. Evelo, Richard Finkers, Alejandra Gonzalez-Beltran, Alasdair J.G. Gray, Paul Groth, Carole Goble, Jeffrey S. Grethe, Jaap Heringa, Peter A.C. t Hoen, Rob Hooft, Tobias Kuhn, Ruben Kok, Joost Kok, Scott J. Lusher, Maryann E. Martone, Albert Mons, Abel L. Packer, Bengt Persson, Philippe Rocca-Serra, Marco Roos, Rene van Schaik, Susanna Assunta Sansone, Erik Schultes, Thierry Sengstag, Ted Slater, George Strawn, Morris A. Swertz, Mark Thompson, Johan Van Der Lei, Erik Van Mulligen, Jan Velterop, Andra Waagmeester, Peter Wittenburg, Katherine Wolstencroft, Jun Zhao, and Barend Mons. Comment: The FAIR Guiding Principles for scientific data management and stewardship. *Scientific Data*, 3 (1):1–9, mar 2016. ISSN 20524463. doi: 10.1038/sdata.2016.18.

Bengt Winblad, Philippe Amouyel, Sandrine Andrieu, Clive Ballard, Carol Brayne, Henry Brodaty, Angel Cedazo-Minguez, Bruno Dubois, David Edvardsson, Howard Feldman, Laura Fratiglioni, Giovanni B. Frisoni, Serge Gauthier, Jean Georges, Caroline Graff, Khalid Iqbal, Frank Jessen, Gunilla Johansson, Linus Jönsson, Miia Kivipelto, Martin Knapp, Francesca Mangialasche, René Melis, Agneta Nordberg, Marcel Olde Rikkert, Chengxuan Qiu, Thomas P. Sakmar, Philip Scheltens, Lon S. Schneider, Reisa Sperling, Lars O. Tjernberg, Gunhild Waldemar, Anders Wimo, and

Henrik Zetterberg. Defeating Alzheimer's disease and other dementias: A priority for European science and society, apr 2016. ISSN 14744465.

Michał Woźniak, Manuel Graña, and Emilio Corchado. A survey of multiple classifier systems as hybrid systems. *Information Fusion*, 16(1):3–17, mar 2014. ISSN 15662535. doi: 10.1016/j.inffus.2013.04.006.

6 Appendix

A Results un-learned ensembles fusing algorithms BLV, BSVM, EMCEB and ResNet

Table 8: BCA and mAUC scores of individual algorithms, un-learned and learned ensembles in the separate training set scenario obtained with 100 times bootstrapping on X_{Test}. Note that ResNet was not used in the learned ensembles.

Separate training set scenario					
Individual algorithms	**BCA**	**mAUC**	*Un-learned ensembles*	**BCA**	**mAUC**
BLV	0.763	0.756	*MeanAll*	0.805	0.890
BSVM	0.768	0.798	*MeanBest*	0.816	0.911
EMCEB	0.761	0.866	*MedianAll*	0.799	0.907
ResNet	0.785	0.896	*MedianBest*	0.820	0.911

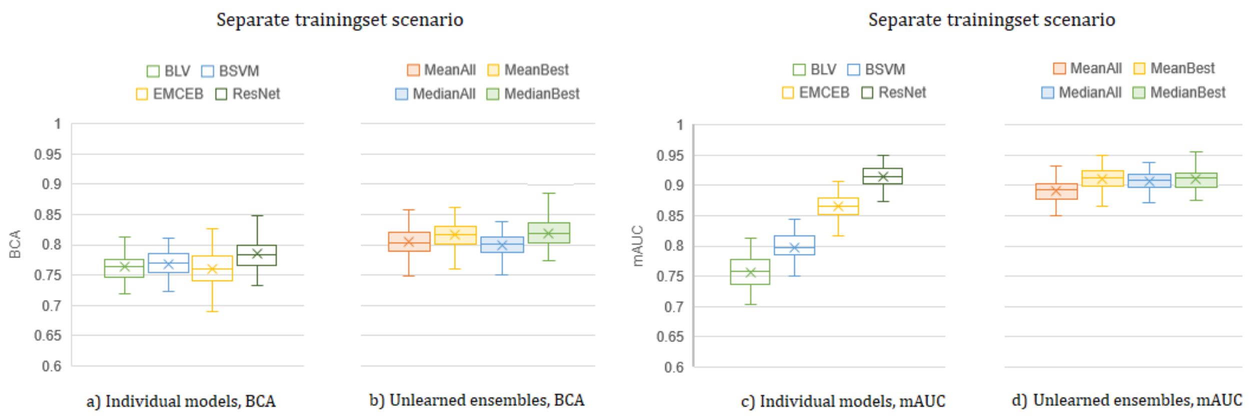


Figure 14: Boxplots of the BCA (a,b) and mAUC (c,d) scores obtained with 100x bootstrapping on X_{Test}. The plots show individual models (non-filled boxplots (a,c)) and un-learned ensemble models (filled boxplots (b,d)). Observe that all of these models were trained according to the separate trainingset scenario. MeanAll - the ensemble with the mean fuser of all four retrainable individual algorithms. MeanBest - the best performing ensemble with the mean fuser on X_{Test} (ensemble of BSVM and ResNet). MedianAll - the ensemble with the median fuser of all four retrainable individual algorithms with the median fuser. MedianBest - the best performing ensemble with the median fuser on X_{Test} (ensemble of BSVM and ResNet).

Table 9: BCA and mAUC scores (means) of experiment 1 where algorithms are trained in the single training set scenario. The performance of the six individual algorithms are also in the table. Observe that the table is ranked on mAUC score.

Model	**BCA**	**mAUC**
Mean_EMC1_ResNet	0.82	0.908
Mean_EMC1_BTMTY_ResNet	0.806	0.905
Mean_EMCEB_EMC1_ResNet	0.814	0.904
Mean_EMCEB_EMC1_BTMTY_ResNet	0.805	0.903
Mean_EMCEB_EMC1	0.782	0.901
Mean_BSVM_EMCEB_BTMTY_ResNet	0.809	0.9
Mean_BSVM_EMCEBEMC1_BTMTY_ResNet	0.802	0.899
Mean_BSVM_EMC1_ResNet	0.809	0.899
Mean_EMCEB_BTMTY_ResNet	0.808	0.899
ResNet	0.838	0.898
Mean_BTMTY_ResNet	0.801	0.898
EMC1	0.792	0.898
Mean_BSVM_EMCEB_EMC1_ResNet	0.805	0.898
Mean_EMCEB_ResNet	0.814	0.897
Mean_EMCEB_EMC1_BTMTY	0.802	0.896
Mean_EMC1_BTMTY	0.805	0.895
Mean_BSVM_EMCEB_BTMTY_ResNet	0.802	0.894
Mean_BSVM_EMCEB_EMC1_BTMTY	0.805	0.893
Mean_BSVM_BTMTY_ResNet	0.803	0.892
Mean_BLV_BSVM_EMCEB_EMC1_BTMTY_ResNet	0.805	0.892
Mean_BSVM_EMC1_BTMTY	0.805	0.892
Mean_EMCEB_BTMTY	0.785	0.892
Mean_BLV_BSVM_EMC1_BTMTY_ResNet	0.809	0.892
Mean_BSVM_EMCEB_EMC1	0.792	0.891
Mean_BLV_EMCEB_EMC1_BTMTY_ResNet	0.802	0.89
Mean_BLV_EMCEB_BTMTY_ResNet	0.805	0.89
Mean_BSVM_EMCEB_ResNet	0.805	0.89
Mean_BLV_EMC1_ResNet	0.803	0.889
Mean_BSVM_ResNet	0.76	0.888
Mean_BLV_EMCEB_EMC1_ResNet	0.811	0.888
Mean_BLV_BSVM_EMCEB_EMC1_ResNet	0.805	0.888
Mean_BSVM_EMCEB_BTMTY	0.792	0.887
EMCEB	0.778	0.886
Mean_BLV_BSVM_EMC1_ResNet	0.806	0.885
Mean_BLV_EMCEB_EMC1	0.781	0.885
Mean_BLV_BSVM_EMCEB_BTMTY_ResNet	0.802	0.885
Mean_BLV_EMCEB_BTMTY_ResNet	0.812	0.885
BTMTY	0.803	0.884
Mean_BSVM_EMC1	0.774	0.884

Mean_BLV_BSVM_EMCEB_EMC1_BTMTY	0.799	0.883
Mean_BLV_EMCEB_EMC1_BTMTY	0.802	0.882
Mean_BLV_BTMTY_ResNet	0.806	0.882
Mean_BLV_BSVM_BTMTY_ResNet	0.796	0.882
Mean_BLV_EMCEB_ResNet	0.794	0.881
Mean_BLV_ResNet	0.76	0.881
Mean_BSVM_BTMTY	0.767	0.88
Mean_BLV_BSVM_EMCEB_EMC1	0.799	0.88
Mean_BLV_EMC1_BTMTY	0.799	0.879
Mean_BLV_EMC1	0.76	0.879
Mean_BLV_BSVM_EMC1_BTMTY	0.799	0.879
Mean_BLV_BSVM_EMCEB_ResNet	0.806	0.878
Mean_BLV_EMCEB_BTMTY	0.792	0.878
Mean_BLV_BSVM_EMCEB_BTMTY	0.799	0.877
Mean_BLV_BSVM_ResNet	0.782	0.875
Mean_BSVM_EMCEB	0.767	0.874
Mean_BLV_EMCEB	0.76	0.873
Mean_BLV_BSVM_EMC1	0.782	0.872
Mean_BLV_BTMTY	0.76	0.867
Mean_BLV_BSVM_EMCEB	0.77	0.867
Mean_BLV_BSVM_BTMTY	0.776	0.867
Mean_BLV_BSVM	0.76	0.82
BSVM	0.767	0.797
BLV	0.76	0.741

Table 10: BCA and mAUC scores (means) of experiment 2 where algorithms are trained in the single training set scenario. The performance of the six individual algorithms are also in the table. Observe that the table is ranked on mAUC score.

Model	**BCA**	**mAUC**
Median_BLV_EMCEB_EMC1_BTMTY_ResNet	0.802	0.913
Median_BLV_BSVM_EMCEB_EMC1_BTMTY_ResNet	0.805	0.911
Median_EMCEB_EMC1_BTMTY	0.799	0.91
Median_BLV_BSVM_EMCEB_BTMTY_ResNet	0.809	0.91
Median_BLV_EMCEB_ResNet	0.799	0.909
Median_BLV_BSVM_EMCEB_EMC1_BTMTY	0.805	0.909
Median_BLV_EMCEB_EMC1_ResNet	0.811	0.909
Median_BLV_BSVM_EMCEB_EMC1_ResNet	0.805	0.909
Median_BSVM_EMCEB_EMC1_BTMTY_ResNet	0.802	0.909
Median_EMC1_ResNet	0.82	0.908
Median_EMCEB_EMC1_BTMTY_ResNet	0.802	0.908
Median_EMCEB_EMC1_ResNet	0.811	0.908
Median_BLV_EMC1_ResNet	0.803	0.907
Median_EMC1_BTMTY_ResNet	0.799	0.907
Median_BLV_EMC1_BTMTY_ResNet	0.805	0.906
Median_BLV_BSVM_EMCEB_ResNet	0.806	0.905
Median_BLV_BSVM_EMC1_BTMTY_ResNet	0.809	0.905
Median_BSVM_EMCEB_EMC1_ResNet	0.805	0.905
Median_BLV_EMCEB_EMC1_BTMTY	0.808	0.905
Median_EMCEB_BTMTY_ResNet	0.812	0.904
Median_BSVM_EMC1_BTMTY_ResNet	0.809	0.903
Median_BSVM_EMC1_ResNet	0.805	0.903
Median_BLV_EMCEB_BTMTY_ResNet	0.808	0.902
Median_BSVM_EMCEB_EMC1_BTMTY	0.805	0.902
Median_BSVM_EMCEB_EMC1	0.799	0.901
Median_BLV_BSVM_EMCEB_EMC1	0.799	0.901
Median_BSVM_EMCEB_BTMTY_ResNet	0.799	0.901
Median_BLV_BSVM_EMC1_ResNet	0.806	0.901
Median_BSVM_EMCEB_ResNet	0.811	0.901
Median_EMCEB_EMC1	0.782	0.901
Median_BSVM_BTMTY_ResNet	0.806	0.899
ResNet	0.838	0.898
Median_BTMTY_ResNet	0.801	0.898
EMC1	0.792	0.898
Median_BLV_EMCEB_EMC1	0.789	0.898
Median_EMCEB_ResNet	0.814	0.897
Median_BLV_EMC1_BTMTY	0.805	0.896
Median_BLV_BSVM_BTMTY_ResNet	0.796	0.896
Median_EMC1_BTMTY	0.805	0.895
Median_BLV_BTMTY_ResNet	0.803	0.894
Median_EMCEB_BTMTY	0.785	0.892
Median_BLV_BSVM_EMCEB_BTMTY	0.799	0.891
Median_BLV_BSVM_EMC1_BTMTY	0.799	0.89

Median_BSVM_EMC1_BTMTY	0.805	0.89
Median_BSVM_EMCEB_BTMTY	0.799	0.888
Median_BSVM_ResNet	0.76	0.888
EMCEB	0.778	0.886
BTMTY	0.803	0.884
Median_BSVM_EMC1	0.774	0.884
Median_BLV_EMCEB_BTMTY	0.798	0.884
Median_BLV_ResNet	0.76	0.881
Median_BSVM_BTMTY	0.767	0.88
Median_BLV_EMC1	0.76	0.879
Median_BSVM_EMCEB	0.767	0.874
Median_BLV_EMCEB	0.76	0.873
Median_BLV_BSVM_EMCEB	0.77	0.871
Median_BLV_BSVM_ResNet	0.782	0.869
Median_BLV_BTMTY	0.76	0.867
Median_BLV_BSVM_EMC1	0.782	0.867
Median_BLV_BSVM_BTMTY	0.776	0.862
Median_BLV_BSVM	0.76	0.82
BSVM	0.767	0.797
BLV	0.76	0.741

Table 11: BCA and mAUC scores (means) of experiment 1 where algorithms are trained in the separate training set scenario. The performance of the four individual algorithms are also in the table. Observe that the table is ranked on mAUC score.

Model	**BCA**	**mAUC**
Mean_EMCEB_ResNet	0.817	0.911
Mean_BSVM_ResNet	0.8	0.904
Mean_BSVM_EMCEB_ResNet	0.817	0.903
ResNet	0.785	0.899
Mean_BLV_ResNet	0.76	0.894
Mean_BLV_EMCEB_ResNet	0.805	0.893
Mean_BLV_BSVM_EMCEB_ResNet	0.805	0.892
Mean_BLV_BSVM_ResNet	0.784	0.891
EMCEB	0.763	0.866
Mean_BSVM_EMCEB	0.767	0.863
Mean_BLV_EMCEB	0.76	0.859
Mean_BLV_BSVM_EMCEB	0.764	0.857
Mean_BLV_BSVM	0.76	0.821
BSVM	0.767	0.798
BLV	0.76	0.741

Table 12: BCA and mAUC scores (means) of experiment 2 where algorithms are trained in the separate training set scenario. The performance of the four individual algorithms are also in the table. Observe that the table is ranked on mAUC score.

Model	**BCA**	**mAUC**
Median_EMCEB_ResNet	0.817	0.911
Median_BLV_EMCEB_ResNet	0.798	0.908
Median_BLV_BSVM_EMCEB_ResNet	0.798	0.907
Median_BSVM_ResNet	0.8	0.904
Median_BSVM_EMCEB_ResNet	0.811	0.902
ResNet	0.785	0.899
Median_BLV_ResNet	0.76	0.894
Median_BLV_BSVM_ResNet	0.784	0.879
EMCEB	0.763	0.866
Median_BSVM_EMCEB	0.767	0.863
Median_BLV_EMCEB	0.76	0.859
Median_BLV_BSVM_EMCEB	0.764	0.846
Median_BLV_BSVM	0.76	0.821
BSVM	0.767	0.798
BLV	0.76	0.741

B Hyper parameter optimalisation for models with a ResNet architecture.

Table 13: Hyperparameter optimisation for models that use the ResNet architecture. LR - learning rate

Initial LR	**Batch size**	**Nodes per layer**	**# of layers**	**% drop out**
0.001	10	50	20	0.05
5E-4	10	50	10	0.05
5E-4	20	50	15	0.05
5E-4	15	100	10	0.05
5E-4	10	20	10	0.05