A deep learning approach for feature extraction from resting state functional connectivity of stroke patients and prediction of neuropsychological scores

---

Candidate:

Delfina Iriarte

External Supervisors:

Dr Alberto Testolin

Prof Marco Zorzi

Internal Supervisors:

Prof. Samir Suweis

Master Degree in Physics of Data

Department of Physics and Astronomy

Università degli Studi di Padova.

April 12, 2022

A deep learning approach for feature extraction from resting state functional connectivity of stroke patients and prediction of neuropsychological scores, April 2022

Author:
Delfina Iriarte

Supervisors:
Prof. Samir Suweis
Dr Alberto Testolin
Prof Marco Zorzi

Institute:
Universita Degli Studi di Padova

ACKNOWLEDGMENTS

*I would like to take this opportunity to thanks the **Università Degli Studi di Padova** for believe in me and gives me this unique opportunity by offering me a generous scholarship. These past years filled me with amazing people from all parts of the world with unforgettable memories.*

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Deep learning models are being increasingly used in precision medicine thanks to their ability to provide accurate predictions of clinical outcome from large-scale datasets of patient's records. However, in many cases data scarcity has forced the adoption of simpler (linear) feature extraction methods, which are less prone to over-fitting. In this work, we exploit data augmentation and transfer learning techniques to show that deep, non-linear autoencoders can in fact extract relevant features from resting state functional connectivity matrices of stroke patients, even when the available data is modest. In particular, we used the Human Connectome Project (HCP) which is a large and high-quality dataset to learn latent representation of healthy patients. The latent representations extracted by the autoencoders can then be given as input to regularized regression methods to predict neurophsychological scores, outperforming recently proposed methods based on linear feature extraction. Additionally, we study the impact of the cross validation set-up for each model, and we examined the quality of the predictive maps obtained by back-projecting the regression weight, to display the most predictive RSFC edges.

Keywords— Resting state networks, Functional connectivity, Deep learning, Feature extraction, Predictive modeling, Neurophysiological Score

1

INTRODUCTION

The rise of neuroimaging in the last years has provided physicians and radiologist with the ability to study the brain with unprecedented ease. **Resting-state functional magnetic resonance imaging (RSfMRI)** is a widely used neuroimaging tool that measures spontaneous fluctuations in neural blood oxygen-level dependent (BOLD) signal across the whole brain in the absence of any controlled experimental paradigm. fMRI data is a commonly used technique by cognitive scientists for investigating the brain activity patterns during different visual tasks and discovering the mechanisms underlying many neurological diseases [1].

Analyses of RSfMRI data have demonstrated temporal correlations in the blood oxygen level–dependent (BOLD) signal of widely separated brain regions. These so-called **resting-state functional connectivity** (RSFC) networks are posited to reflect intrinsic representations of functional systems commonly implicated in cognitive function [2]. One important goal of current neuroimaging research is to associate individual RSFC with behavioral scores. However, establishing relationships between resting-state brain activity and cognitive or clinical scores is still a difficult task, in particular in terms of prediction as would be meaningful for clinical applications [3]. Research efforts in fMRI are shifting focus from studying specific cognitive domains like vision, language, memory, and emotion to assessing individual differences in neural connectivity across multiple whole-brain networks [4]. Siegel, Ramsey, Snyder, Metcalf, Chacko, Weinberger, Baldassarre, Hacker, Shulman, and Corbetta [5] show that visual memory and verbal memory deficits are better predicted by functional connectivity than by lesion location, and visual and motor deficits are better predicted by lesion location than functional connectivity.

One fundamental issue, often found in this studies, is the so-called **small-n-large-p**. The number of subjects frequently ranges from tens to hundreds, whereas the number of features (namely voxels) to be analysed can add up to millions. This negatively affects the statistical power of any experiment performed [6, 7], since without pre-selecting the 'most relevant' features and effectively discarding redundant features plus noise, a predictive machine learning model has a marked risk of 'overfitting'. Therefore, a fundamental step before applying a model in neuroimaging studies is to reduce the dimensionality of the data. With the arrival of the deep learning paradigm, it has become possible to extract high-level

abstract features directly from MRI images that internally describe the distribution of data in low-dimensional manifolds.

Machine Learning (ML) is one of the most exciting and rapidly expanding fields within computer science, that it have gained prominence for the analysis of RSfMRI data providing an alternative analytical approach for estimating neuroanatomical alterations [8]. Rather than being explicitly programmed for a certain task, machine learning systems are able to find relevant data, discover patterns and predict the outcome of the input data [6]. Unsupervised machine learning methods have proven promising for the analysis of high-dimensional data with complex structures, making it evermore relevant to rs-fMRI. A vast majority of literature on machine learning for rs-fMRI is devoted to unsupervised learning approaches. Unlike task-driven studies, modelling resting-state activity is not straightforward since there is no controlled stimuli driving these fluctuations [8].

As already mentioned, Rs-fMRI data is highly dimensional and several feature extraction method can be perform in order to reduce the dimensionality of the data. Such limitation can be partially addressed by exploiting linear dimensionality reduction techniques such as Principal Component Analysis (PCA), Independent Component Analysis (ICA), or sparse coding in combination with regularized regression methods. Previous studies has already been implemented in order to address this problem: Calesella, Testolin, De Filippo De Grazia, and Zorzi [9] studied the impact of regularization in combination with different dimensionality reduction techniques, to establish which method can be more effective to build predictive models of behavioral outcome from RSFC of patients with stroke. Nevertheless, the choice of the dimensionality reduction technique is non-trivial because it can affect performance of the predictive model ([9, 10]).

In particular, PCA works by finding the direction of the greatest variance in the dataset and represents each data point by its coordinates along each of these directions [11]. However, this method is essentially a linear transformation and cannot extract nonlinear structures modeled by higher than second-order statistics [12]. In order to overcome the nonlinear dimensionality reduction, an autoencoder (AE) can be implemented which can learn non-linear transformations with a non-linear activation function and multiple layers. AEs are a neural network based alternative for generating reduced feature sets through nonlinear input transformations. They have been used for feature reduction of RS-FC in several studies [8]. AEs can also be used in a pre-training stage for supervised neural network training, in order to direct the learning towards parameterspaces that support generalization. This technique was shown, for example, to improve classification performance of Autism and Schizophreniausing RSFC [13, 14]. Notice that if the autoencoder is comprised of a simple fully connected encoder and decoder with a squared loss objective, it performs dimension reduction equivalent to PCA. However, the non-linearity activation functions often allows for a superior reconstruction when compared to simple PCA.

Previous studies using autoencoder as a feature extraction method for psychiatric neuroimaging have already been implemented. Pinaya, Mechelli, and Sato [15] extracted features from structural MRI scans of the HCP dataset using a convolutional AE; Huang, Hu, Zhao, Makkie, Dong, Zhao, Li, and Liu [16] use a deep convolutional autoencoder to extract high level features of **task**-fmri data using the HCP dataset; GENG and Xu [17] used an autoencoder as a dimensionality reduction process using rs-FMRI data set of patients with major depressive disorder (MDD).

The application of deep learning to build accurate predictive models from functional neuroimaging data is often hindered by limited dataset sizes, which is often referred in terms of ML as **curse of dimensionality**. One way to deal with this problem is to used data augmentation techniques to increase the number of images in our dataset. Another solution is to implement transfer learning techniques, that allows one to take knowledge learned about one deep learning problem and apply it to a different, yet similar learning problem.

In this work, we show that better performance can be achieved by exploiting the representational power of non-linear dimensionality reduction techniques, namely, deep autoencoders [18]. Nevertheless, the application of such powerful deep learning models is often hindered by the limited size of clinical datasets. In this work we propose to mitigate this issue using two complementary approaches: data augmentation, which allows to significantly expand the sample size by combining/distorting existing samples, and transfer learning, which allows to exploit additional large-scale datasets (in our case, from the Human Connectome Project [19]) containing functional connectivity data in order to pre-train the autoencoder.

The proposed approach is validated on a reference dataset containing functional connectivity matrices of stroke patients [5]. The features extracted by the autoencoder are used as predictors of the corresponding neurophsychological scores (language, spatial memory and verbal memory) by means of regularized linear regression methods. The latter can limit multicollinearity and overfitting, which makes them particularly suitable for the analysis of neuroimaging data (for a recent review, see [20]).The performance of our method is benchmarked against other popular dimensionality reduction methods based on PCA, ICA and sparse coding, showing promising results. Additionally, the quality of the predictive maps obtained by back-projecting the regression weight to display the most predictive RSFC edges are examined for every model. Finally, the impact of the augmentation of the dataset are further studied.

2

THEORY

In this section, basic concepts of resting-state fMRI are review. Afterwards, the theory of the models used as features extraction for the stroke rs-FMRi dataset are presented. Specifically, we will present the theory of Principal Component Analysis (PCA), Independent component Analysis (ICA) and several types of autoencoders (AE). Finally, the regularized regression used as predictors of the language behavioral scores are introduced.

2.1 RESTING-STATE FMRI

Magnetic resonance imaging (MRI) is a non-invasive medical imaging technique that consisted on measuring the differences in resonances of different materials while in the presence of a strong magnetic field. In the setting of neuroscience, MRI is often used to image the inside of the brain while it is performing basic functions like hearing a tone, or pressing a button, or even at rest. This flavor of MRI is appropriately referred to as **Functional MRI, or fMRI** [21].

Functional magnetic resonance imaging (fMRI) is a non invasive technique for measuring brain activity with an excellent spatial resolution (few millimeters for an imager 3 T). This method essentially measures the differences in resonances of brain tissue based on functionally-dependent levels of blood oxygen. This is commonly referred to as the Blood-Oxygen-Level-Dependent (BOLD) signal (Figure 2.1).

At the microscopic level, a neuron consists of a cell body (soma) that receives input through dendrites and passes action potentials through their axons to other cells. These microscopic processes in turn result in a localized increase in blood flow leading to an increase in local oxygen consumption in brain tissue, slightly increasing the concentration of deoxyhemoglobin in blood. Oxygenated and deoxygenated blood has different magnetic properties. As a result, the above causes a BOLD signal increase. Thus, the magnitude of the BOLD signal change therefore depends on the change in the concentration of deoxyhemoglobin that results from the imbalance between the increase in oxygen consumption and the increase in blood flow [22] (Figure 2.2). To help us interpret changes in resting state functional connectivity, it is useful to have an understanding of how increases in excitatory input to a population of neurons lead to a localized increase in blood flow. The

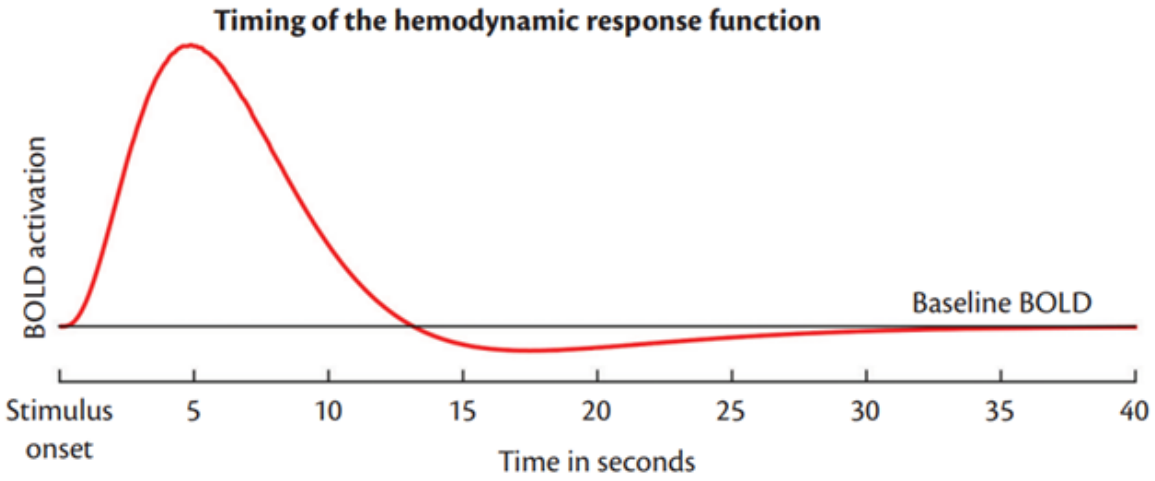


Figure 2.1: The BOLD signal is an indirect measure of neuronal activity that is mediated by a slow increase in local oxygenated blood flow that takes several seconds to peak. The standard form of the hemodynamic response function is shown. From stimulus onset, the BOLD signal takes approximately 5 seconds to reach its maximum (taken from [23]).

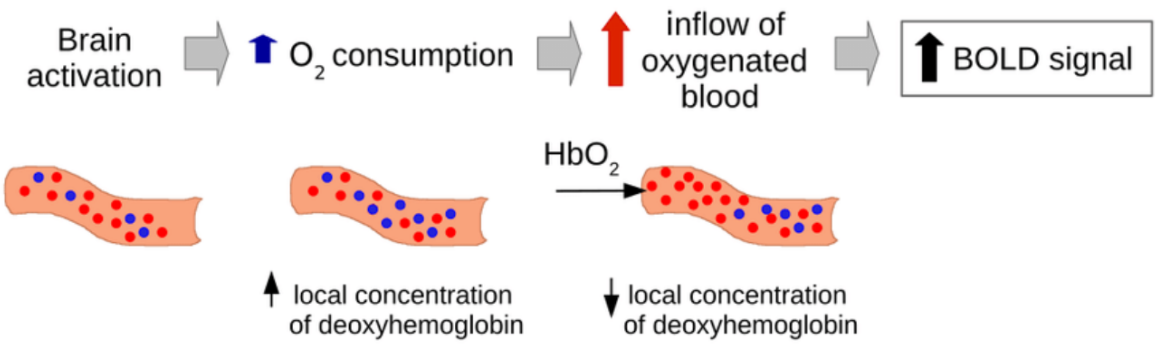


Figure 2.2: Hemodynamic effects contributing to the BOLD signal during activation (taken from [21])

active process that links this postsynaptic activity to increased blood flow and volume to the region is called neurovascular coupling, and this process mediates the signal measured in resting state fMRI. Therefore, neurovascular coupling is a term that describes the effects of several complex chemical processes that involve all the different types of cells that exist in a population of neurons [23, 24].

fMRI data is acquired by repeated imaging of the brain while the subject or patient executes a task or receives a sensory stimulus during repeated epochs separated by periods of rest. This data is analyzed by correlating the measured time-varying BOLD signal in each image location with a predicted BOLD signal, obtained by convolving the known function representing the stimulus with a Hemodynamic Response Function (HRF) modeling the delay in the vascular response. The general idea of fMRI signal processing is depicted in Figure

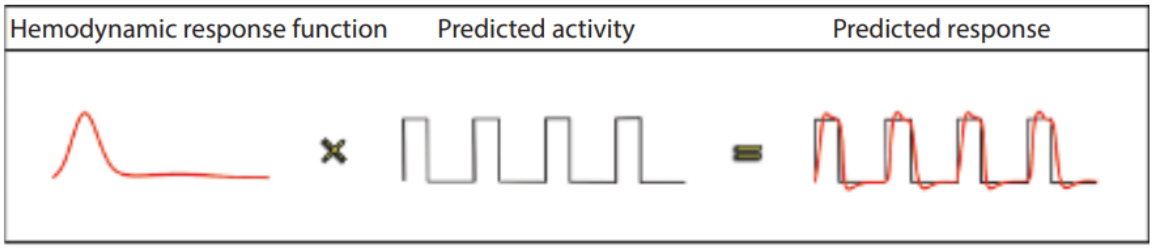


Figure 2.3: Convolution of the predicted activity curve of the fMRI experiment with a hemodynamic response function (HRF), producing the so-called predicted response (Taken from [25])

2.3: the process starts by convolving the predicted activity curve of the fMRI experiment, with a hemodynamic response function (HRF), producing the so-called predicted response. Each voxel contains a time-varying BOLD signal. Signals that match the predicted response (that is the modeled change in the BOLD signal) are identified as activation related to stimulus and can be processed for statistical analysis [25]. However, before entering the statistical analysis, it has to be ensured that the data is artifact and noise free. Hence several preprocessing steps has to be done [25]:

1. slice scan timing correction,
2. head motion correction,
3. distortion correction
4. spatial and temporal smoothing of the data

Locations in the brain where this correlation is statistically significant are considered to exhibit a neuronal response to the task or stimulus, and thus to be involved in its cognitive processing [23, 24].

Functional connectivity is typically defined as the observed temporal correlation (or other statistical dependencies) between two neurophysiological measurements from different parts of the brain. For resting state fMRI this definition means that functional connectivity can inform us about the relationship between BOLD signals obtained from two separate regions of the brain. Parcel can be used as nodes to approximate brain networks as graphs. The underlying assumption is that if two regions show similarities in their BOLD signals over time, they are functionally connected. Many different methods exist to look at such similarities, the simplest way to investigate similarity between two signals is by looking at their timeseries correlation using Pearson's correlation coefficient. Correlation ranges from -1 (perfect negative correlation) to $+1$ (perfect positive correlation), where 0 indicates no relationship on average between two signals. Therefore, the connectivity is the way in which brain regions communicate with one another and information is passed from one brain area to the next. To investigate it, the similarity of the BOLD signals is measured from different brain regions, because if the signals are similar, this is likely to mean that the regions are passing on information from one region to the other (i.e., there is connectivity) [23, 24].

In order to study connectivity, we often look at spontaneous fluctuations in the signal, when there are no specific cognitive demands for the subject (so-called resting state scans). Using spontaneous fluctuations allows us to investigate similarity between regions when it is not biased by any specific task. As such, resting state fMRI has emerged as a valuable way to study brain connectivity [23, 24]. The procedure of RSfMRI for the subject is relatively easy, and in any case non-demanding, compared to task fMRI since only remaining calm inside the MRI scanner for about 10 minutes is required, trying not to think anything in particular[22].

2.2 FEATURE EXTRACTION METHODS

For a given dataset with n data points or records and p features, high−dimensional data is observed when the number of features p is higher than the number of records n, as $p > n$ [26]. Neuroimaging data is highly dimensional i.e. it has a large number of features in each image. Some studies use whole feature set from brain images, but a common practice is to reduce the number of features since many features may not contribute. As a result, feature reduction techniques are used to remove redundant predictor variables and experimental noise, a process which mitigates the curse-of-dimensionality and small-n-large-p effects [27].

Dimensionality reduction is the process of taking data in a high dimensional space and mapping it into a new space whose dimensionality is much smaller [28]. This is an essential step before training a machine learning model to avoid overfitting and therefore improving model prediction accuracy and generalization ability.

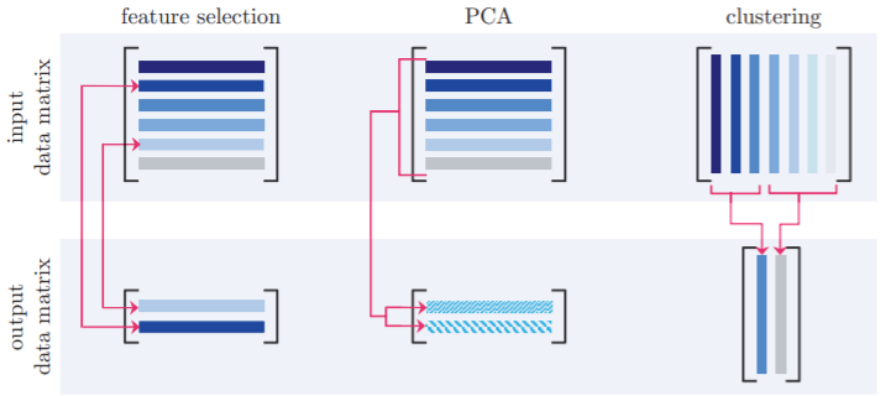


Figure 2.4: Comparison of feature selection, PCA, and clustering as dimension reduction schemes on an arbitrary data matrix. The former two methods reduce the dimension of the feature space, or in other words the number of rows in a data matrix. However, the two methods work differently (taken from [29])

There are several ways to reducing the data dimension of a dataset (Figure 2.4):

1. **Feature selection:** Feature selection is a process in which a subset of the features'space is chosen according to its relevance to the output of the classifier. The ultimate goal is to extract the most effective subset of features and to remove redundant irrelevant information. [26].
2. **Feature extraction:** Feature-extraction or feature-reduction techniques identify a new subset of features that are transformed or combined from the original feature space to obtain a more significant set of features. Feature-extraction methods can be linear or non-linear. Principal component analysis (PCA) is one such common technique.
3. **Clustering:** reduces the dimension of the data/number of data points, or equivalently the number of columns in the input data matrix. It does so by finding a small number of new averaged representatives or "centroids" of the input data, forming a new data matrix whose fewer columns (which are not present in the original data matrix) are precisely these centroids.

In the present work, we are going to focus only on **features extraction** techniques such us Principal Component Analysis (PCA) and Independent Component Analysis (ICA).

Lowering the number of features to reduce computational effort is an essential concept in feature extraction. In these methods, the reduction is performed by applying a **linear transformation** to the original data. That is, if the original data is in $\mathcal{R}^d$ and we want to embed it into $\mathcal{R}^n$ $(n < d)$ then we would like to find a matrix $W \in \mathcal{R}^{n,d}$ that induces the mapping $x \to Wx$. A natural criterion for choosing W is in a way that will enable a reasonable recovery of the original x [28]. It should be point out that these techniques are unsupervised, therefore no label are required, which is a common scenario in many rs-fmri images.

Nowadays, modern deep architectures can be used that allow determining latent features, while reducing the number of features for further processing and keeping the relevant information at the same time. In particular, we are interested in undercomplete autoencoders that learn a representation z which is smaller than the original input x. This architecture is contrary to overcomplete autoencoders, in which the representation of the data is higher dimensional than the number of input features. Also, undercomplete autoencoders are more common, because once we learned the compressed representation, the computational effort for further processing is reduced in comparison to the original input and the overcomplete representation [30]. On the other hand, autoencoders are **non-linear** and **can learn more complicated relations** between visible and hidden units. They can also be stacked, which makes them even more powerful.

In the following, the theory of linear features extraction techniques, such as Principal Component analysis and Independent component analysis, as well as non-linear techniques are introduced in depth.

2.2.1 *Principal Component analysis*

Principal Component analysis (PCA) is feature extraction technique meant to reduce the dimensions of the dataset. The compression and the recovery are performed by linear transformations and the method finds the linear transformations for which the differences between the recovered vectors and the original vectors are minimal in the least squared sense [28].

Let $x_1, x_2, ...x_m$ be m vectors in $\mathcal{R}^d$. The aim is to find the recovery matrix $W \in \mathcal{R}^{n,d}$ where $n < d$ that induces a lower dimensionality representation of x by mapping $x \to Wx \in \mathcal{R}^n$. In PCA, the compression matrix W and the recovering matrix $U \in \mathcal{R}^{d,n}$ are founded so that the total squared distance between the original and recovered vectors is minimal; namely, we aim at solving the problem:

$$\underset{W \in \mathcal{R}^{n,d}, U \in \mathcal{R}^{d,n}}{\operatorname{argmin}} \sum_{i=1}^{m} ||x_i - UWx_i||_2^2 \tag{2.1}$$

In particular, the solutions (U, W) of Equation 2.1 are such that the columns of U are orthonormal and $W = U^T$, therefore we can rewrite it as:

$$\underset{U \in \mathcal{R}^{d,n}: U^T U = I}{\operatorname{argmin}} \sum_{i=1}^{m} ||x_i - UU^T x_i||_2^2 \tag{2.2}$$

Equation 2.2 can be rewritten after some elementary algebraic manipulation as:

$$\underset{U \in \mathcal{R}^{d,n}: U^T U = I}{\operatorname{argmax}} \operatorname{trace}\left(U^T \sum_{i=1}^{m} x_i x_i^T U \right) \tag{2.3}$$

Let $A = \sum_{i=1}^{m} x_i x_i^T$ be the covariance matrix, and let $u_1, ..., u_n$ be $n$be n eigenvectors of the matrix A corresponding to the largest n eigenvalues of A. Then, the solution to the PCA optimization problem given in Equation 2.3 is to set U to be the matrix whose columns are $u_1, ..., u_n$ and to set $W = U^T$. The eigenvectors and eigenvalues of a covariance (or correlation) matrix represent the "core" of a PCA: The eigenvectors (principal components) determine the directions of the new feature space, and the eigenvalues determine their magnitude. In other words, the eigenvalues explain the variance of the data along the new feature axes.

2.2.2 *Independent Component Analysis*

Independent component analysis (ICA) is an algorithm that performs decomposition imposing that the resulting components must be independent. Let $x_1, x_2, ...x_n$ be n observable

variables. We assume that they can be modelled as linear combinations of hidden (latent) variables $s_j, j = 1, ...m$ with some unknown coefficients a_{ij}:

$$x_i(t) = \sum_{j=1}^{m} a_{ij} s_j \qquad \forall i = 1, .., n \tag{2.4}$$

The fundamental point is that we observe only the variables x_i, whereas both the coefficients, a_{ij}, and the independent components, s_i, are to be estimated or inferred. The main breakthrough in the theory of ICA was the realization that the model can be made identifiable by making the unconventional assumption of the non-Gaussianity of the independent components.

2.2.2.1 *Fast ICA*

Entropy is the basic concept of information theory. The entropy of a random variable can be interpreted as the degree of information that the observation of the variable gives. The more "random", i.e. unpredictable and unstructured the variable is, the larger its entropy. In this work, we will use the FastICA algorithm. FastICA is a block fixed point iteration algorithm based on negative entropy, or negentropy, as a non-gaussianity measure. Fixed-point algorithms converge faster than adaptive algorithms [6]. For a random vector y, negentropy can be defined as:

$$J(y) = H(y_{gauss}) - H(y) \tag{2.5}$$

where y_{gauss} is a Gaussian random variable of the same covariance matrix as y, and H is the entropy function. Negentropy is always non-negative, and it is zero if and only if y has a Gaussian distribution The FastICA defines negentropy using the function:

$$J(y) \sim [E\{G(y)\} - E\{G(\nu)\}]^2 \tag{2.6}$$

where we assume that y is of zero mean and unit variance, ν is a Gaussian variable sharing the same mean and variance, and $G(x)$ can be any non-quadratic function. Many functions have been proposed, common choices are $G(x)_1 = (1/a_1)log(cosh(a_1 x))$ with $1 < a_1 < 2$ or $G(x)_2 = exp(-x^2/2)$ [31]. With these measures, we can compute the derivatives of these functions by:

$$g_1(x) = tanh(a_1 x) \tag{2.7}$$

$$g_2(x) = xexp(-x^2/2) \tag{2.8}$$

The algorithm for the one-unit version of FastICA can be defined [31] as:

1. Choose an initial (e.g. random) weight vector w.
2. Let $w^+ = E\{xg(w^T x)\} - E\{g'(w^T x)\}w$
3. Let $w = w^+/||w^+||$
4. If not converged, go back to 2

The algorithm considers that the values of w converge when their dot product is close to 1, that is, they are pointing in the same direction.

2.2.3 *Autoencoder*

The autoencoder is an unsupervised neural-network based approach for learning latent representations of high-dimensional data. It encodes the input data into a lower dimensional representation (latent space), which is then decoded to reconstruct the input.

It consists of two parts:

1. **Encoder**: It translates the original high-dimension input into the latent low-dimensional code. The input size is larger than the output size.
2. **Decoder**: The decoder network recovers the data from the code, likely with larger and larger output layers.

Traditionally, autoencoders were used for dimensionality reduction or feature learning [32]. An autoencoder whose code dimension is less than the input dimension is called undercomplete. Learning an undercomplete representation forces the autoencoder to capture the most salient features of the training data [33]. The idea is that the bottleneck serves as a feature extractor of the input data.

Let's consider a basic auto-encoder with a single hidden layer, n neurons in the input/output layers and m neurons in the hidden layer. The model takes an input $\mathbf{x} \in \mathcal{R}^n$ and first maps it into the latent representation $\mathbf{h} \in \mathcal{R}^m$ by using an encoding function $\mathbf{h} = g_\phi(\mathbf{x}) = \sigma(W\mathbf{x} + b)$ with parameters $\phi = \{W, b\}$, where $\sigma(\cdot)$ denotes the activation function of the neurons, W denotes the connection weights and b denotes the neurons' biases. Afterwards, a reconstruction of the input $\mathbf{x}'$ is obtained through the decoder function $\mathbf{x}' = f_\theta(\mathbf{h}) = \sigma(W'\mathbf{h} + b')$ with $\theta = \{W', b'\}$, as shown in Figure 2.5.

The two parameter sets are usually constrained to be of the form $W \in \mathcal{R}^{n,m} = W^T \in \mathcal{R}^{m,n}$ (weight sharing), using the same weights for encoding the input and decoding the latent representation [35]. The parameters (θ, ϕ) are learned together to output a reconstructed data sample same as the original input, $\mathbf{x} \approx f_\theta(g_\phi(\mathbf{x}))$, or in other words, to learn an identity function. There are various metrics to quantify the difference between two vectors, such as cross entropy when the activation function is sigmoid, or as simple as MSE loss:

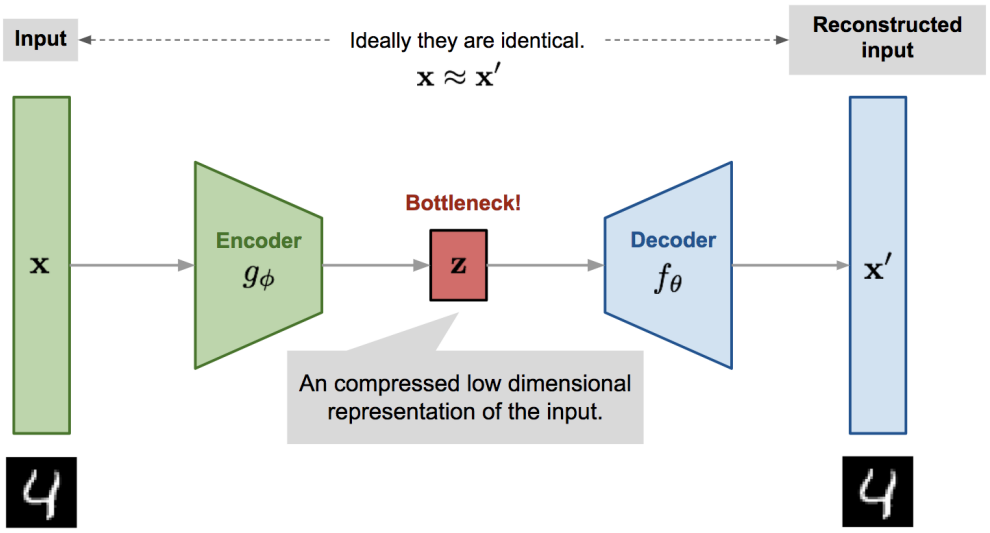


Figure 2.5: Undercomplete AE: The AE reduces the dimensionality in each layer of the encoder (Taken from [34])

$$L_{\text{AE}}(\theta, \phi) = \frac{1}{n} \sum_{i=1}^{n} (\mathbf{x}^{(i)} - f_\theta(g_\phi(\mathbf{x}^{(i)})))^2 \tag{2.9}$$

The autoencoder can be seen as a non-linear extension of PCA. In fact, a simple autoencoder which consist of only one linear layer with a linear activation function is compared with the same one with a non-linear activation function and a PCA model [35]. Autoencoders with nonlinear encoder functions and nonlinear decoder functions can thus learn a more powerful nonlinear generalization of PCA.

2.2.4 *Convolutional Autoencoders (CAE)*

Fully connected AE ignore the image structure. **Convolutional Autoencoder (CAE)** is a variant of Convolutional Neural Networks that are used as the tools for unsupervised learning of convolution filters. They are generally applied in the task of image reconstruction to minimize reconstruction errors by learning the optimal filters. CAE differs from conventional AEs as their weights are shared among all locations in the input, preserving spatial locality [35]. The reconstruction is hence due to a linear combination of basic image patches based on the latent code. For a mono-channel input x the latent representation of the k$-th$ feature map is given by [35]:

$$h^k = \sigma(x * W^k + b^k) \tag{2.10}$$

where the bias is broadcasted to the whole map, σ is an activation function, and $*$ denotes a convolution. The reconstruction is obtained using:

$$y = \sigma(\sum_{k \in H} h^k * \hat{W}^k + c) \tag{2.11}$$

where c reprent the bias per input channel. H identifies the group of latent feature maps; $\hat{W}$ identifies the flip operation over both dimensions of the weights [35]. The cost function to minimize is the mean squared error (MSE):

$$E(\sigma) = \frac{1}{2n} \sum_{i=1}^{n} (x_i - y_i)^2 \tag{2.12}$$

The parameters W and b are founded by minimizing the loss function. In order to learn hierarchical latent representations, several AEs can be stacked together thus forming a deep neural network [35].

2.2.5 *Autoencoder: Overcomplete case*

An overcomplete autoencoder is an autoencoder which has an equal or larger number of hidden units (latent space z) than input units. This type of network architecture gives the possibility of learning greater number of features, but on the other hand, it has potential to learn the identity function and become useless [36]. Additional constraints must be imposed so that most hidden units are enforced to be close to zero. Recently, it has been observed that when representations are learnt in a way that encourages sparsity, improved performance is obtained on classification tasks [37]. Additionally, the sparsity constraint is expected to be advantageous in this context because it encourages representations that may disentangle the underlying factors controlling the variability of MRI images [36].

In particular, two sparsity approaches are implemented in this work:

1. L_1 **normalization:** Regularized autoencoder instead of preserving the encoder and decoder shallow and the code size small, uses a loss function that encourages the network to prevent from just copy its input to its output. The loss function worked by penalizing activations of hidden layers so that only a few nodes are encouraged to activate when a single sample is fed into the network. L1 regularization adds "absolute value of magnitude" of coefficients and tends to shrink the penalty coefficient to zero.
2. **k-sparse autoencoder** [37]: in the latent space only the k highest activities are kept, rather than reconstructing the input from all of the hidden units. This selection step acts as a regularizer that prevents the use of an overly large number of hidden units [37].

2.3 REGULARIZED REGRESSION

Consider the usual linear regression model, given p predictors $x_1, ..., x_p$, the response y is predicted by:

$$\hat{y} = \hat{\beta}_0 + \boldsymbol{x_1}\hat{\beta}_1 + ... + \boldsymbol{x_p}\hat{\beta}_p \tag{2.13}$$

A model fitting procedure produces the vector of coefficients $\hat{\boldsymbol{\beta}} = \left(\hat{\beta}_0, ..., \hat{\beta}_p\right)$. For example, the ordinary least squares (OLS) estimates are obtained by minimizing the residual sum of squares [38]. Therefore its objective is to find the plane that minimizes the sum of squared errors (SSE) between the observed and predicted response:

$$\min\left\{SSE = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2\right\} \tag{2.14}$$

Under "nice" condition, OLS works fine, but with real data, OLS often does poorly in both prediction and interpretation. Penalization techniques have been proposed to improve it in which now a penalty parameter (P) a penalty term is added in the objective function:

$$\min\left\{SSE + P\right\} \tag{2.15}$$

Regularized regression puts constraints on the magnitude of the coefficients and will progressively shrink them towards zero. Ridge regression [39] is an example that controls the coefficients by adding $\lambda\sum_{j=1}^{p}\beta_j^2$ (known also as Tikhonov regularization) to the objective function. The least absolute shrinkage and selection operator (lasso) model [40], is an alternative to ridge regression that has a small modification to the penalty in the objective function. Rather than the L_2 penalty, the L_1 penalty $\lambda\sum_{j=1}^{p}|\beta_j|$ is used in the objective function. Whereas the ridge regression approach pushes variables to approximately but not equal to zero, the lasso penalty will actually push coefficients to zero.

Although the lasso has shown success in many situations, it has some limitations. For example, in the "large p, small n" case (few samples n compared to many predictors p), the LASSO selects at most n variables before it saturates. Also if there is a group of highly correlated variables, then the LASSO tends to select one variable from a group and ignore the others. To overcome these limitations, we can make use of a generalization of the ridge and lasso models which is the **elastic net** [41], which combines the two penalties:

$$\min_{(\beta_0,\boldsymbol{\beta})}\left(\boldsymbol{y} - \beta_0 - \boldsymbol{X}^T\boldsymbol{\beta}\right)^2 + \lambda\left(\frac{1}{2}\left(1-\alpha\right)\boldsymbol{\beta}^2 + \alpha\left|\boldsymbol{\beta}\right|\right), \tag{2.16}$$

Therefore, the elastic-net loss function requires two free parameters to be set, namely the λ and α parameters. The parameter α controls the type of shrinkage, with important consequences for the properties of the estimation method. The penalty parameter λ controls the amount of shrinkage [42]. Lasso ($\alpha = 1$) has an l_1 penalty on the parameters and performs both parameter shrinking and variable selection. The other end, $\alpha = 0$, gives Ridge regression with a l_2 penalty on the parameters, which does not have the variable selection property.

3

MATERIALS AND METHODS

3.1 DATASETS

The main dataset used in our study consists of 132 resting-state functional connectivity (RSFC) matrices from symptomatic stroke patients, taken from previous studies [5, 9]. The patients underwent a 30-minute-long RS-fMRI acquisition, 1–2 weeks after the stroke occurred. Several scores were taken during the neuropsychological assessment: here we focus on language, verbal memory and spatial memory indexes, which are available for a subset of 119 subjects. In this dataset, twenty-one patients were excluded for hemodynamic lags and 11 patients and 4 controls were excluded for excessive head motion [5]. After exclusion, 100 stroke patients and 27 age-matched controls were studied. RSFC data represent the connectivity between brain regions that share functional properties and can be expressed as a symmetric matrix. In our case, the matrix of each subject is of size 324×324; following common practice [9], the data is vectorized by only considering the upper triangular matrix. Null values were converted to zero.

In order to implement a transfer learning approach, we also used as dataset the Human Connectome Project [19], consisting of RSFC matrices of 1050 healthy subjects.

In the following subsections, a brief discussion about the pre-processing of the data implemented by Siegel, Ramsey, Snyder, Metcalf, Chacko, Weinberger, Baldassarre, Hacker, Shulman, and Corbetta is presented.

3.1.1 *Neuropsychological assessment for the stroke dataset [5]*

The behavioral scores were obtained from previous studies done by Siegel, Ramsey, Snyder, Metcalf, Chacko, Weinberger, Baldassarre, Hacker, Shulman, and Corbetta [5], in which they provide scores for the motor, language, attention, memory, and visual domain. The experimental set-up that they have done was the same for all patients. In the language domain, the Boston diagnostic aphasia examination (BDAE) test was used in order to examine the comprehension, expression and reading skills. In the memory domain, the Brief Visuospatial Memory Test (BVMT) was used to asses the spatial recall and recognition test and the Hopkins Verbal Learning Test (HVLT) to asses verbal skills.

3.1.2 *Parcellation (Regions of Interest) for the stroke dataset [5]*

The cortical parcellation was also obtained from previous studies [5] and it was based on Gordon, Laumann, Adeyemo, Huckins, Kelley, and Petersen. The parcellation includes 324 regions of interest (159 left hemisphere, 165 right hemisphere) (Figure 3.1). The original parcellation includes 333 regions, and all regions less than 20 vertices (approximately 50 mm2) were excluded.

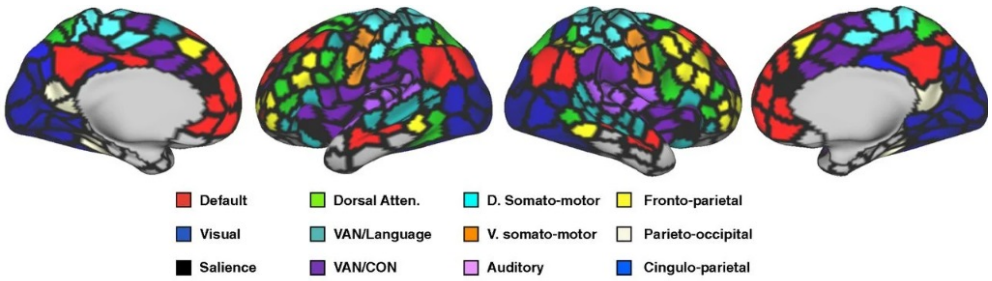


Figure 3.1: The 324 region on interest parcellation from [43]. Regions are color coded by RSN membership (Taken from [5]).

3.1.3 *Functional Connectivity processing for the stroke dataset [5]*

The Functional connectivity matrices from [5], were obtained by computing between each parcel using Fisher z-transformed Pearson correlation 3.2. The Fisher Z-Transformation is a way to transform the sampling distribution of Pearson's so that it becomes normally distributed. Connectivity for any parcel that fell within the boundaries of the lesion was removed from univariate analyses and set to zero for multivariate models.

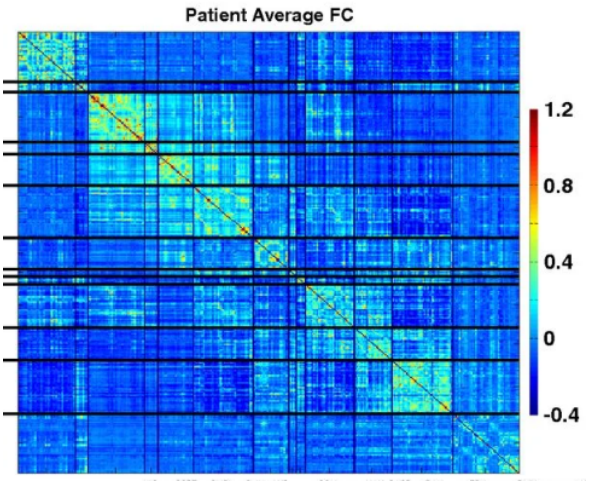


Figure 3.2: Average Fisher z-transformed FC matrices are shown for stroke patients excluding regions that overlap lesions (Taken from [5]).

3.2 FEATURE EXTRACTION METHODS

Our main focus is to test different variants of deep autoencoders in their ability to extract useful features from RSFC data, and compare their performance with standard linear dimensionality reduction methods [9]. The models are compared based on its reconstruction error which correspond to the mean squared error between the original image and the reconstructed one. During the unsupervised feature extraction process, the entire dataset (n=132) was used regardless of the availability of neuropsychological scores.

3.2.1 *Principal Component analysis*

Prior performing PCA is necessary to standardize the data in order to avoid biased results. This leads with data that is centerized with zero mean and variance one. This is done by using the predefined function `StandardScaler` from SKLEARN that standardize features by removing the mean and scaling to unit variance. Principal Component analysis is performed by using the function `PCA` from the library SKLEARN which performs linear dimensionality reduction using Singular Value Decomposition of the data to project it to a lower dimensional space.

3.2.2 *Independent Component Analysis*

ICA performs the decomposition step by imposing the constraint that the resulting components must be independent. In this work we used the `FastICA` algorithm from SKLEARN, which is a block fixed-point iteration algorithm based on negative entropy as a non-gaussianity measure, which converges faster than adaptive algorithms [6]. As in the case of PCA, data is first standardized.

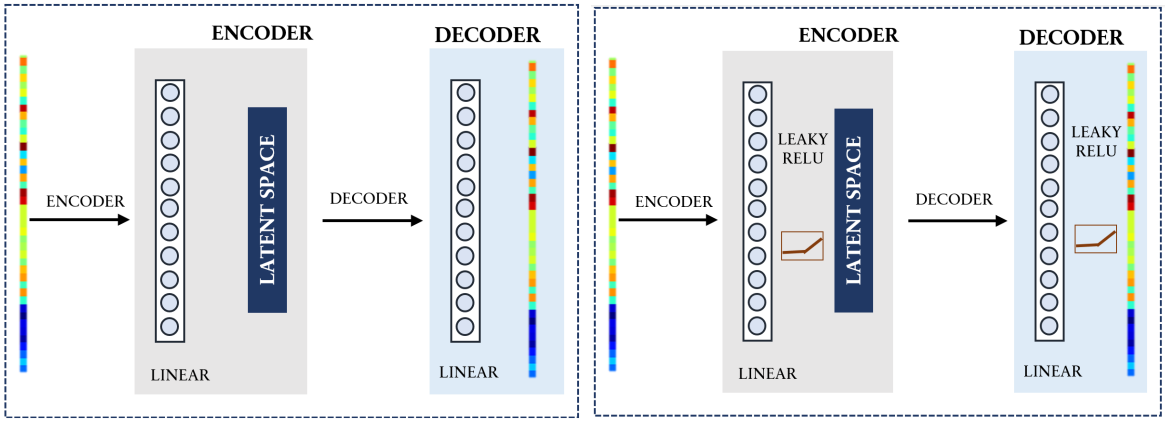
3.2.3 *Autoencoders*

The data is divided intro train, validation and test set by using the predefined function `train_test_split` of SKLEARN. In this work we considered from simple AE to more advance architectures. After training them, the bottleneck layer provides as the features extracted of the data.

3.2.3.1 *One-layer autoencoder*

First of all, a simple one linear layer autoencoder is performed in KERAS as shown in figure 3.3a (**Lin AE**-model). Adam optimizer with learning rate $1e-3$ is used and the loss function is given by the mean squared error setting the `'mse'` flag in the optimizer.

The same model is performed but using LeakyReLU as a non-linear activation function (Figure 3.3b (**NonLin AE**-model)). No hyperparamenter search is performed.



(a) With linear activation function (b) With non linear activation function

Figure 3.3: Model architecture of an architecture consist on one layer.

3.2.3.2 *Convolutional autoencoder*

Figure 3.4 presents the architecture of the Deep Convolutional Autoencoder performed (**CAE**-model), which takes as input the vectorized matrix of the fMRI dataset. As it can be observed, the encoder consist on 3 convolutional layers follow by 2 fully connected ones, whereas the decoder is simply the inverse. In order to overcome vanishing gradient problem and "dying ReLU" problem, the Leaky Rectified Linear activation function was used, allowing models to learn faster and perform better. Mean Square Error was used as loss function and Adam as Optimizer. Furthermore, dropout was used as a regularization.

As framework, the code was written using PyTorch and PyTorch Lightning which is an open-source Python library. Moreover, in order to optimize the hyperparameter search of the model, Optuna [44] was used. Table 3.1 presents the hyperparamter search space. Furthermore, Optuna [44] uses the pruning algorithm. Pruning is a technique used in machine learning and search algorithms to reduce the size of decision trees, by removing sections of the tree that are non-critical and redundant to classify instances. Pruning in Optuna automatically stops unpromising trials at the early stages of the training.

In this work, we consider the overcomplete and the undercomplete case of the convolutional autoencoder. In particular, we use two sparsity approaches:

- **CAE-L1**: Overcomplete convolutional autoencoder with the same architecture as shown in Figure 3.4 with a latent space of 200 and L_1 regularization search-space equal to $0.0001, 0.001, 0.01, 0.1, 1, 4$,

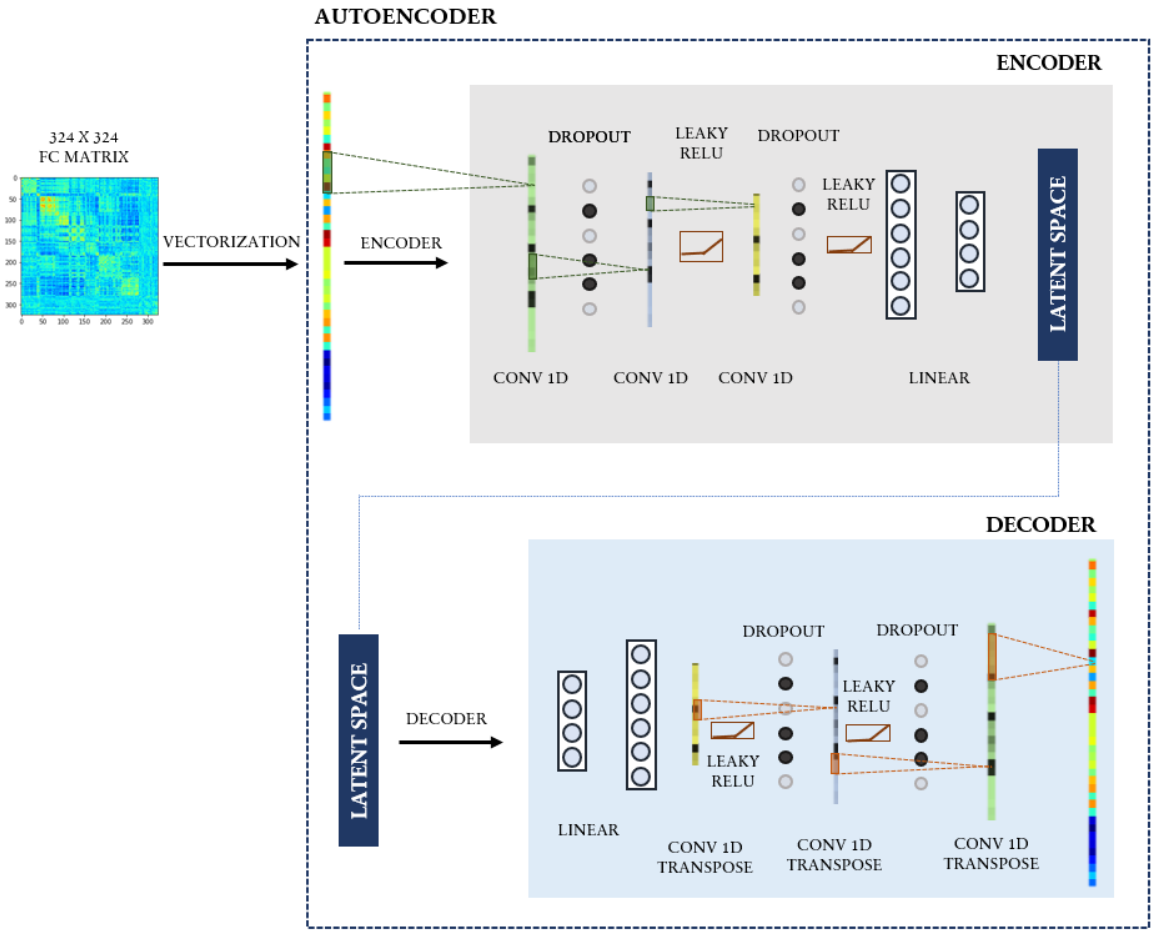


Figure 3.4: Schematic representation of the workflow and the Deep Convolutional Autoencoder used.

- **CAE-k**: Overcomplete convolutional autoencoder with the same architecture as shown in Figure 3.4 with a latent space of 200 using k-sparcity as constrained with search-space $k = 10, 30, 60, 90$.

Curse of dimensionality:

Deep convolutional neural networks performed remarkably well, but these networks are heavily reliant on big data to avoid overfitting. To build useful Deep Learning models, the validation error must continue to decrease with the training error. Given the limited size of our clinical dataset, we thus devised two approaches in order to promote a better generalization of the CAE during the feature extraction process.

Data Augmentation is a very powerful method, which consists in combining and distorting each training sample in order to provide a more representative distribution as input to the autoencoder [45]. The augmented data will represent a more comprehensive set of possible data points, thus minimizing the distance between the training and validation set,

Hyperparameter	Range
Conv1	[8,16,32, 64, 128]
Conv2	[8,16,32, 64, 128]
Conv3	[8,16,32, 64, 128]
fc	[8,16,32, 64, 128]
learning rate	(1e-5 - 1e-4)
dropout	(0 - 1)
weight decay	(1e-5 - 1e-2)

Table 3.1: Hyperparamters search space learned using OPTUNA

as well as any future testing sets [45]. In particular, a mixup augmentation is performed that consist on a random convex combination of raw inputs (**CAE-AUG**-model):

$$\hat{x} = \lambda x_i + (1 - \lambda) x_j$$

where x_i and x_j are raw input vectors and λ are values sampled from the Beta distribution. Following previous work [46], the choice of the parameters $\lambda \in [0, 1]$ was distributed accordingly to $\lambda \in Beta(\alpha, \alpha)$ for $\alpha \in (0, \inf)$. In the mix-up, the samples to be combined are chosen randomly from all available images. Isaksson, Summers, Raimondi, Gandini, Bhalerao, Marvaso, Petralia, Pepa, and Jereczek-Fossa tested the utility of the mix-up data augmentation technique for a medical image segmentation task using 100 MRI scans and observed an improvement when $\alpha = 0.5$. Although our dataset could be slightly different, the decided to use the same α value for consistency. The size of the extended dataset is now ~ 7000

Another solution, is provided by **Transfer Learning (CAE-TL-model)** that describes a process in which a model is trained on one dataset and subsequently applied to another dataset. Transfer learning refers to machine learning techniques that focus on acquiring knowledge from related tasks to improve generalization in the tasks of interest. A DL model that has been pre-trained on a large, openly available fMRI dataset, generally requires less training data and time, and achieves higher decoding accuracies, when compared to a model variant with the same architecture that is trained entirely from scratch. The Human Connectome Project database is used in order to explore the benefits of TL.

The same architecture as shown in Figure 3.4 is implemented. Afterwards, the model is trained using the stroke dataset but keeping freeze the convolutional layers (Figure 3.5).

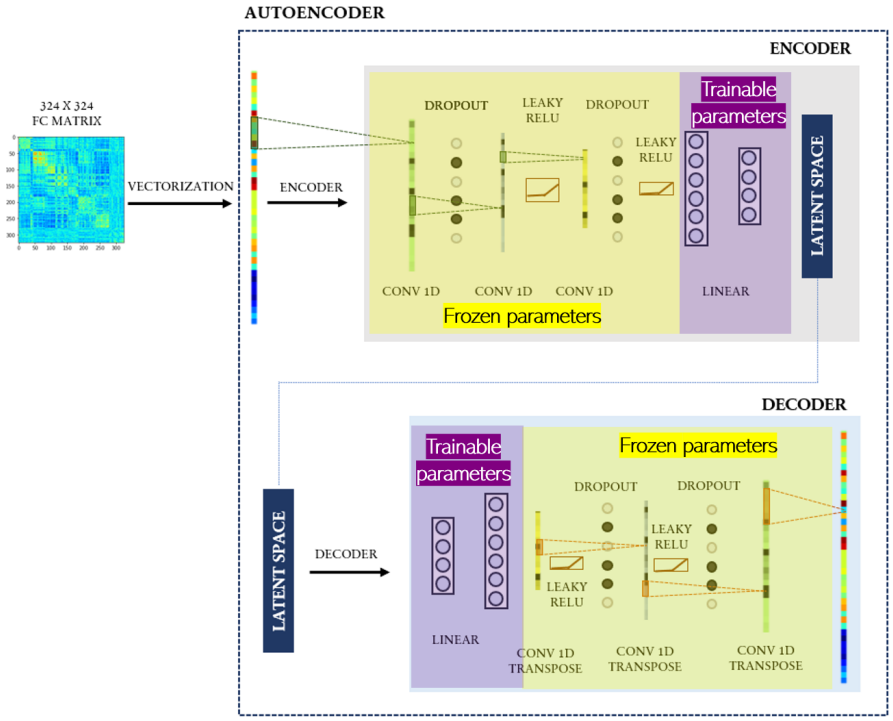


Figure 3.5: Schematic representation of the transfer learning approach.

3.2.4 *Getting deeper on Augmentation techniques:*

Given the remarkable performance of the CAE trained using data augmentation and transfer learning, a series of additional simulations are explored in order to see how the size of the augmented dataset could impact model performance, and whether a combination of augmentation and transfer learning might further improve the accuracy. We thus designed four additional training regimens:

1. **Aug(15000)**: Similarly as before, the CAE is trained with synthetic images obtained via the mix-up strategy; however this time the size of the augmented stroke dataset is increased to ~ 15000 samples (i.e., twice the size used previously);
2. **TL-Aug**: The CAE is first trained over the HCP dataset, as done before for the Transfer Learning scenario. The model is then also trained on the initial augmented stroke dataset (~ 7500 samples);
3. **AugTL-Aug:** The CAE is first trained over synthetic HCP data obtained by applying the same mix-up augmentation strategy (~ 6000 samples). The model is then also trained on the initial augmented stroke data (~ 7500 samples);

4. **AugTL-Stroke:** The CAE is first trained over synthetic HCP data obtained by applying the same mix-up augmentation strategy (~ 6000 samples). The model is then also trained on the original stroke dataset.

It should be point out that in this case there hasn't been an exhaustive hyper-parameter search due to lack of time. Instead of searching the hyper-parameter with 50 trials using OPTUNA, only 5 trials have been done.

3.3 REGULARIZED REGRESSION

The feature sets extracted by each method were then used as regressors for the prediction of the neuropsychological scores. Note that only the subjects with available score were kept in this phase. Recall the optimization function of the ElasticNet in Equation 2.16. The tuning parameters λ and α can be chosen by k-fold cross validation. Moreover, Leave-One-Out Cross validation (LOOCV) was used in where the number of folds equals the number of instances in the data set, therefore at N separate times, the function approximator is trained on all the data except for one point and a prediction is made for that point (Figure 3.6). The combination of hyper-parameters leading to the model with lowest MSE was selected as the "best model".

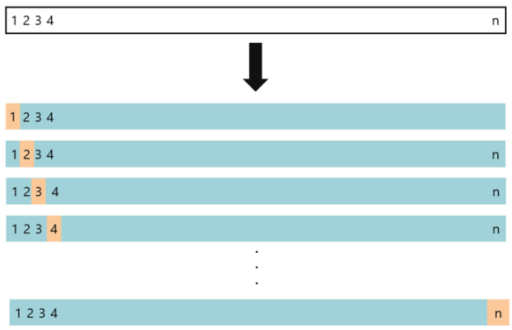


Figure 3.6: Schematic display of LOOCV. A set of n data points is repeatedly split into a training set (shown in blue) containing all but one observation, and a validation set that contains only that observation (shown in beige). The test error is then estimated by averaging the n resulting MSE's. The fistr training set contains all but observation 1, the second training set contains all but observation 3, and so forth. Taken from [48].

3.3.1 *Cross-validation setup*

In the standard LOOVCV, however, selection of the best model is based only on the test error, which could lead to optimistically biased model performance. One approach to overcoming this bias is to nest the hyperparameter optimization procedure under the model

selection procedure. This is called double cross-validation or nested cross-validation and is the preferred way to evaluate and compare tuned machine learning models. The nested CV has an inner loop CV nested in an outer CV. The inner loop is responsible for model selection/hyperparameter tuning (similar to validation set), while the outer loop is for error estimation (test set). The algorithm is as follows [49] (Figure 3.7):

1. Divide the dataset into K cross-validation folds at random.
2. For each fold $k = 1, 2, \cdot, K$: outer loop for evaluation of the model with selected hyperparameter
   a) Let test set be fold k
   b) Let `trainval`-set be all the data except those in fold k
   c) Randomly split `trainval`-set into L folds
   d) For each fold $l = 1, 2, \cdot L$ (**inner loop**):
      i. Let validation-set be fold l
      ii. Let training-set be all the data except those in test or validation set
      iii. Train with each hyperparameter on trainining-set, and evaluate it on validation-set. Keep track of the performance metrics.
   e) For each hyperparameter setting, calculate the average metrics score over the L folds, and choose the best setting.
   f) Train a model with the best hyperparameter on `trainval`. Evaluate its performance on test-set and save the score for fold k.
3. Calculate the mean score over all K folds, and report as the generalization error.

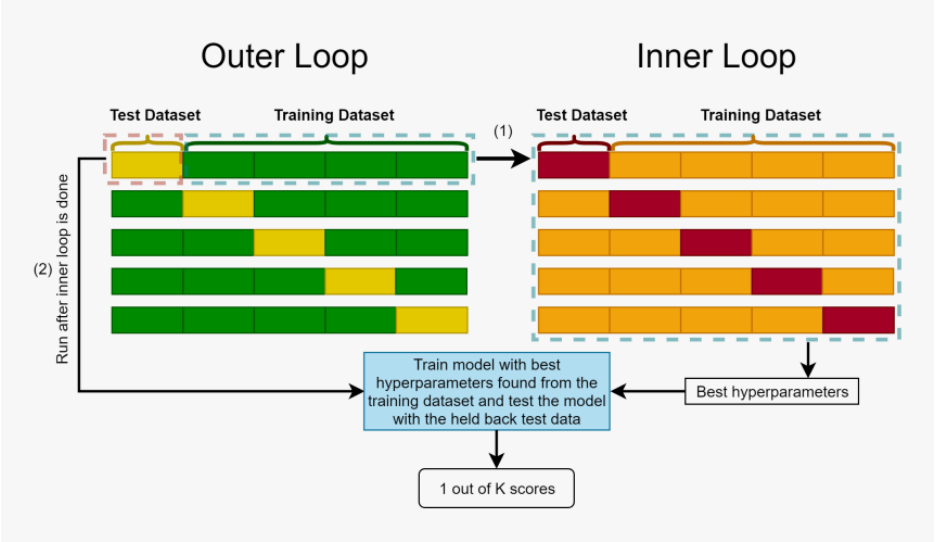


Figure 3.7: Schematic display of Nested cross validation. we have two loops. The inner loop is basically normal cross-validation with a search function. Though the outer loop only supplies the inner loop with the training dataset, and the test dataset in the outer loop is held back. Taken from: https://mlfromscratch.com/nested-cross-validation-python-code/

A drawback of this approach is that it can lead to the choice of different models across the CV loops: to produce the final model of the n-LOO procedure, three measures of central tendency were used for choosing the optimal hyper-parameters, namely mean (n-average condition), median (n-median condition) and mode (n-mode condition) [9].

3.3.2 *Model comparison criterion*

To evaluate the regression model obtained, the most commonly known metrics are used:

1. **R-squared (R2)**: the proportion of variation in the outcome that is explained by the predictor variables. In multiple regression models, R2 corresponds to the squared correlation between the observed outcome values and the predicted values by the model. The Higher the R-squared, the better the model. This is obtained by calculating:

$$R^2 = 1 - \frac{RSS}{TSS} \tag{3.1}$$

   where RSS is the sum of squares of residuals and the TSS is the total sum of squares.
2. **Mean Squared Error (MSE):** defined as Mean or Average of the square of the difference between actual and estimated values
3. **Bayesian information criterion (BIC):** is a criterion for model selection among a finite set of models. It is based, in part, on the likelihood function, and it is closely related to Akaike information criterion (AIC). The BIC statistic is calculated for logistic regression as follows:

$$ln(n)k - 2ln(\hat{L}) \tag{3.2}$$

   being $\hat{L}$ the value of the maximum of the likelihood function, n the number of samples and k the number of parameters in the model.

3.4 BACK-PROJECTING

Finally, for each method, the optimal regression coefficients were back-projected in the original space, by means of linear transformation through the features' weights, and restored in a symmetric matrix for the PCA-model and ICA-model, whereas for the autoencoder models, we use the decoder in order to obtained its back-projection. This provides a map that displays the predictive edges in the resting state networks.

4

RESULTS AND DISCUSSION

4.1 FEATURE EXTRACTION

The feature extraction methods were first assessed based on their reconstruction error, which is given by the mean square error between the original data and the reconstructed one. Since the techniques used are unsupervised, all the dataset (n=132) was used at this stage regardless of the availability of the score.

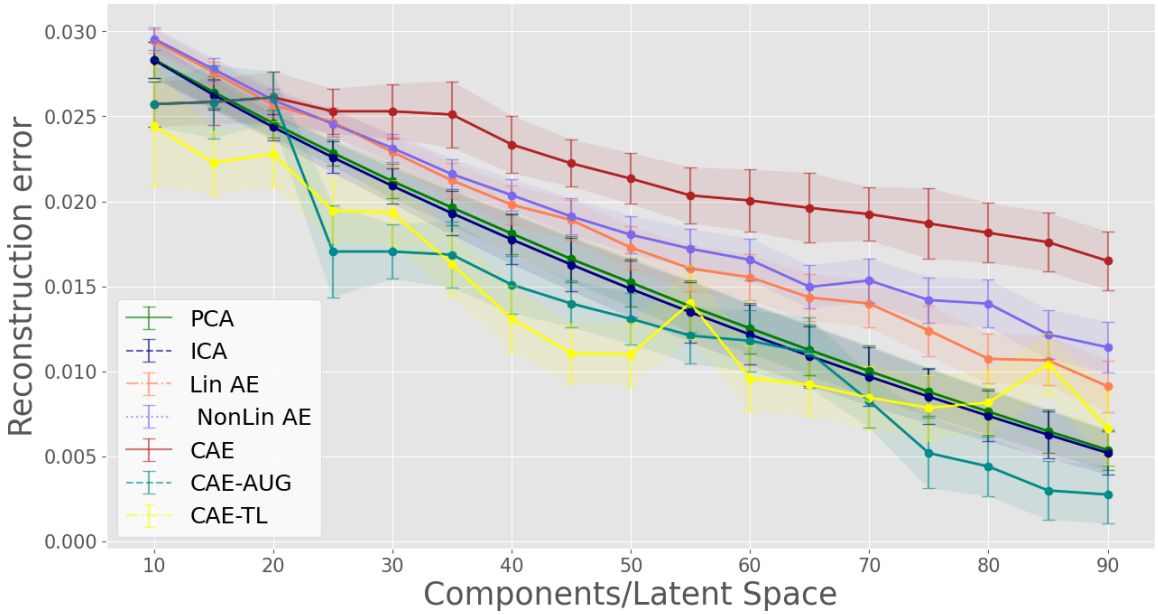


Figure 4.1: Reconstruction error obtained for the several models against the latent space/number of components.

Figures 4.1 shows the reconstruction error against the number of components/latent space. As expected, the larger the number of components/latent space, the better the reconstruction error. In particular it can be observed, that the convolutional autoencoder trained directly to the stroke dataset is the one with the worst reconstruction error. This can be because of the fact that many samples (generally $\mathcal{O}(3)$) are needed to train deep convolutional architectures, highlights the importance of increasing the variability of the training

distribution in order to improve the quality of the features extracted by complex convolutional architectures. Furthermore, Figure 4.2 shows the best and the worst reconstruction samples of the convolutional autoencoder trained directly to the stroke dataset.

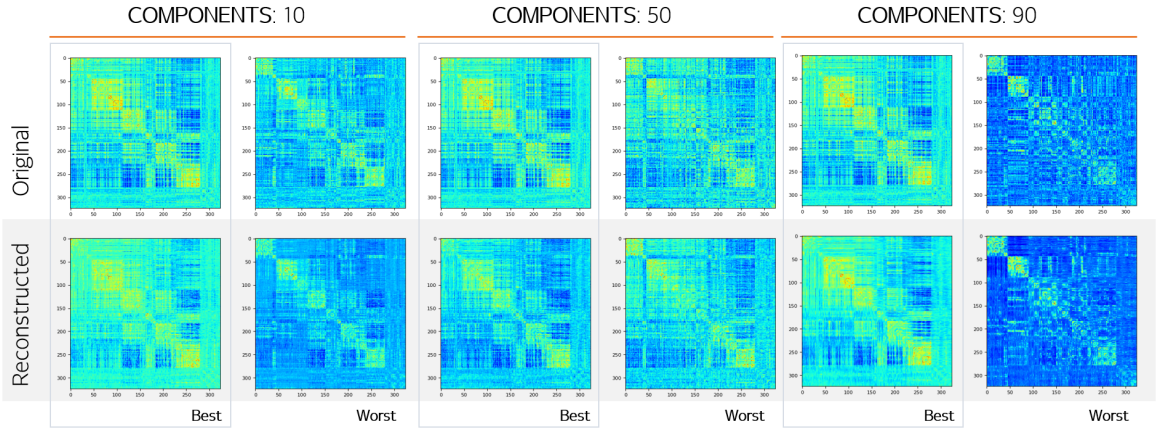


Figure 4.2: Best and worst reconstructed samples of the strokes samples obtained by applying the convolutional autoencoder trained with the original dataset with latent space equals to 10, 50, 90.

Convolutional autoencoder samples applied to the augmentated dataset (obtained by mix-up strategy as explained in Section 3) are shown in 4.3. It should be point out that even if the model was trained with a larger dataset (around ~ 7000 samples generated by augmentation techniques), the reconstruction error shown in Figure 4.1 correspond to the mean square error between the stroke samples only. On the other hand, samples obtained after applying transfer learning are shown in Figure 4.4.

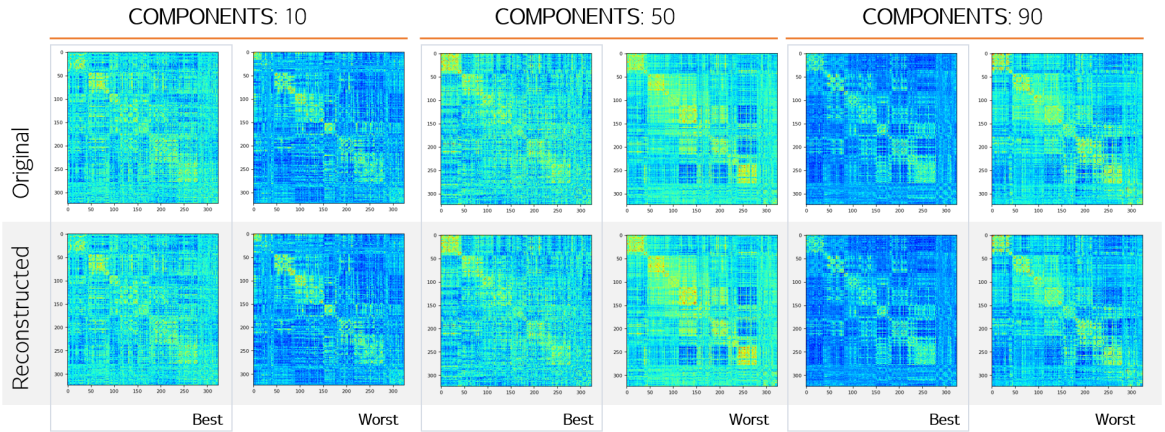


Figure 4.3: Best and worst reconstructed samples of the strokes dataset obtained by the convolutional autoencoder trained with synthetic data with latent space equals to 10, 50, 90.

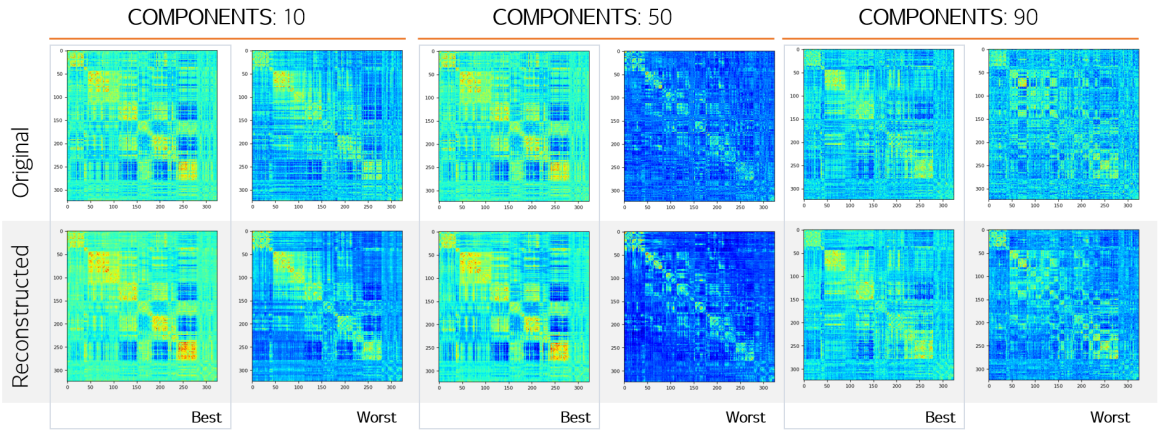


Figure 4.4: Best and worst reconstructed samples of the strokes dataset obtained by the convolutional autoencoder by applying transfer learning using the HCP dataset with latent space equals to 10, 50, 90.

Figures 4.5 and 4.6 presents the original and their correspond reconstructed samples via PCA and ICA respectively with the lowest reconstruction error. In general, it can be seen that the two methods performs quite similar. This assumption can also be observed by noticing that the Reconstruction error behaves similarly as shown in Figure 4.1. In particular, it can be seen that ICA is able to better reconstructed samples with smaller details than the PCA. It should be mentioned that PCA and ICA are much faster than autoencoders.

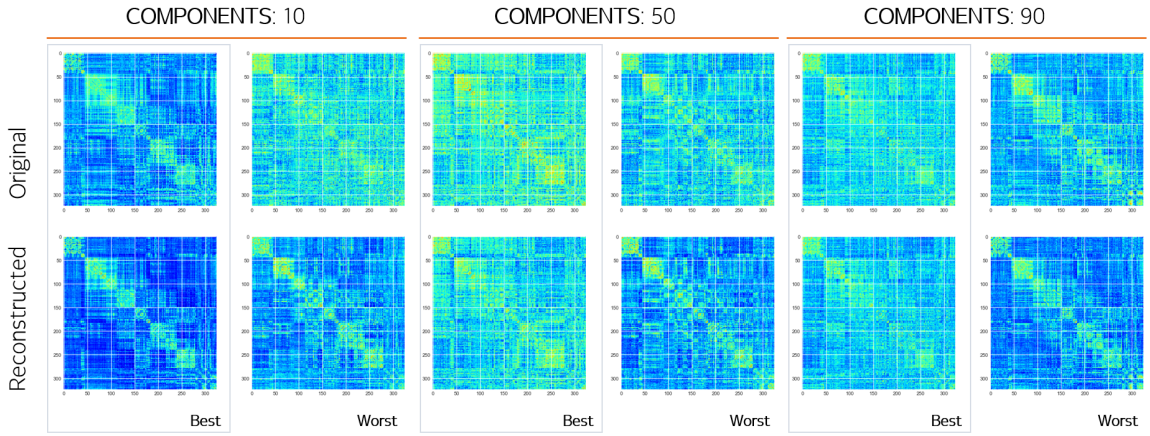


Figure 4.5: Best and worst reconstructed samples obtained by PCA using 10, 50, 90 components.

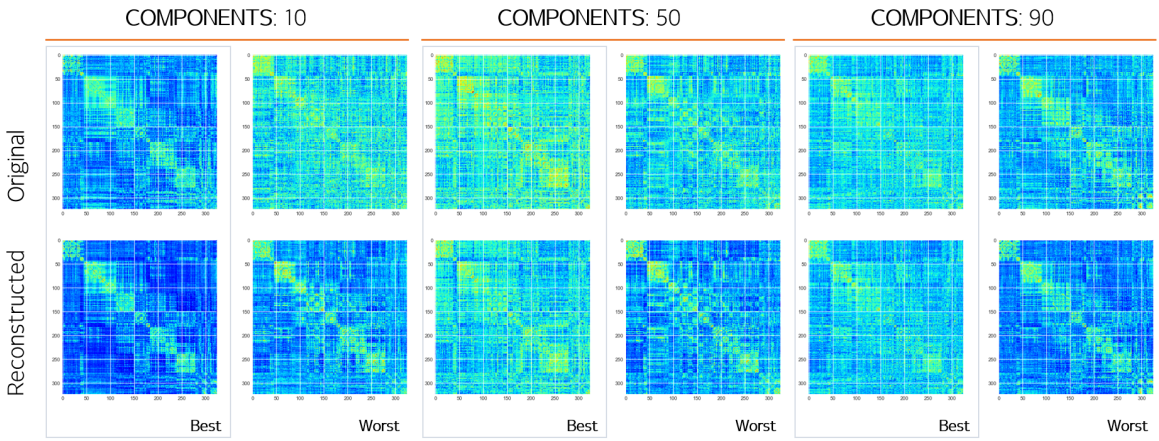


Figure 4.6: Best and worst reconstructed samples obtained by ICA using 10, 50, 90 components.

On the other hand, it should be worth mentioning that the $1-$layer linear AE gives similar reconstruction error as PCA (Figure 4.1) which is no surprising due to the intrinsic similarity between PCA and a simple autoencoder. In particular, it can be observed that even when using the $1-$layer nonlinear AE, the reconstruction errors are consistent with the other methods. Samples of of reconstructed images with the lowest reconstruction error for these two methods are shown in Figure 4.7 and Figure 4.8.

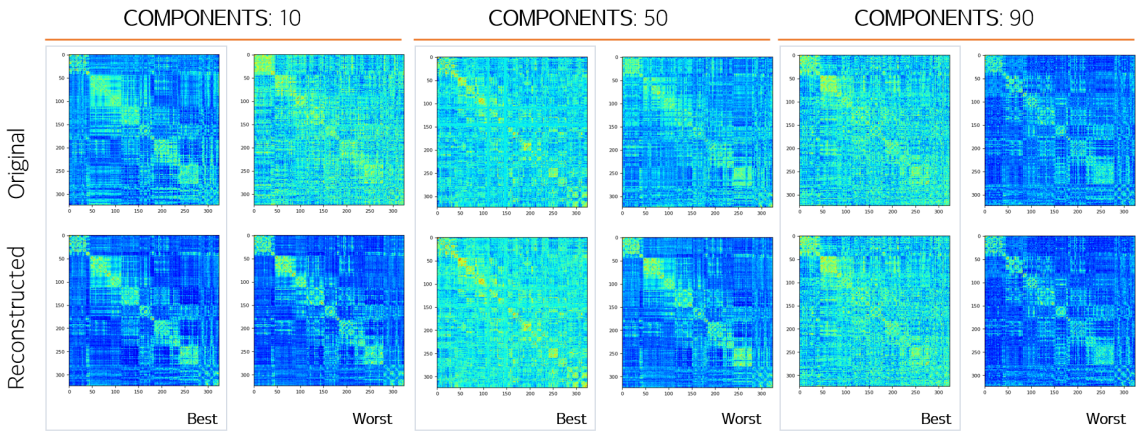


Figure 4.7: Best and worst reconstructed samples obtained by an autoencoder consist of one linear layer with latent space equals to 10, 50, 90.

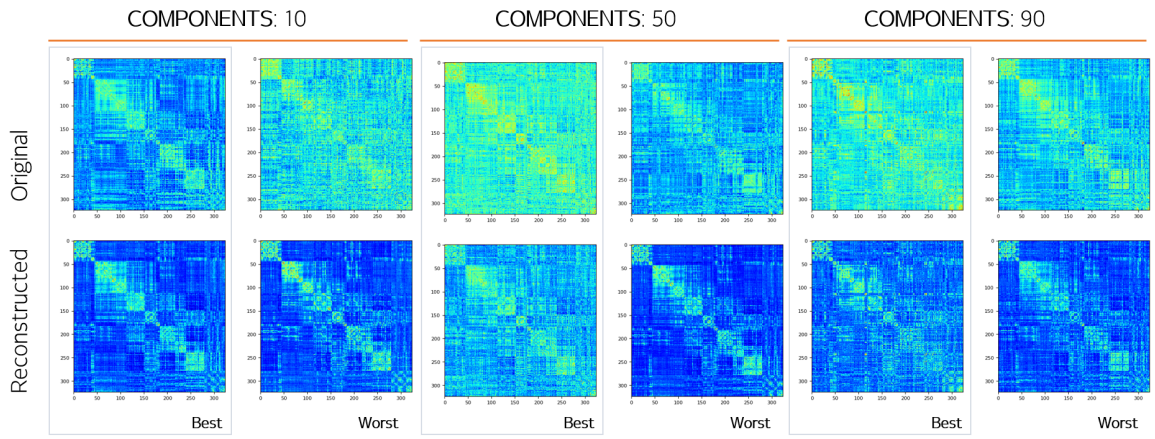


Figure 4.8: Best and worst reconstructed samples obtained by an autoencoder consist of one layer with a non linear activation function (`LeakyReLU`) with latent space equals to 10, 50, 90.

4.2 REGULARIZED REGRESSION

Table 4.1 presents the metrics obtained by predicting the neurophysiological scores available: **Language**, **Spatial Memory**, and **Verbal Memory**, using the traditional and non−traditional methods already mentioned. Notice that each of the features were first standardized, since this highly affected the performance of the model. Additionally, Table 4.2 presents the metrics in the **overcomplete**-regime of the deep-learning methods as explained in Section 2.2.5. Notice that the latent space in this case was 200 which is greater than the available input data for all types of neurophysiological scores.

Concerning the metrics obtained for the language score, it can be observed from Table 4.1, that PCA performs better when comparing with the MSE error and R^2 with all rest models, though the margin is fairly small. However, Table 4.2 shows that the predictions obtained by using the features from the applying transfer learning in the overcomplete case with a k-sparsity constrained were exactly leading the same results as PCA. On the other hand, the convolutional autoencoder applied to the stroke dataset directly is the one leading to the worst results.

In order to have a better visualization, Figure 4.9a presents the methods sorted by lowest MSE error and highest R^2 using language as neurophysiological score. As already stated, $PCA-$based model and overcomplete transfer with k-sparsity were the ones providing the best values. On the other hand, the metrics obtained when using the features from the one-single-layer AE, and the convolutional autoencoder are the worst ones. In particular, it can be noticed that in the overcomplete regime using k-sparsity as constraint leads to better accuracies than using L_1 regularization, same as Makhzani and Frey [37] claimed in their

	Method	R^2	MSE	BIC	Optimal α	Optimal λ	Fold	NZ
Language Score (n=94)	**PCA**	0.522	0.478	492.696	0.001	0.221	65	65
	ICA	0.510	0.490	350.616	0.25	0.087	35	35
	Lin AE	0.428	0.572	322.987	0.25	0.06	20	20
	NonLin AE	0.500	0.504	356.65	0.75	0.004	30	30
	CAE	0.425	0.575	623.703	0.25	0.005	90	90
	CAE-AUG	0.504	0.498	420.68	0.5	0.06	50	43
	CAE-TL	0.444	0.555	453.777	0.001	0.034	50	50
Spatial Score (n=77)	**PCA**	0.214	0.786	299.91	1	0.087	40	23]
	ICA	0.240	0.750	395.69	0.001	0.56	55	55
	Lin AE	0.267	0.733	411.831	0.001	0.559	50	50
	NonLin AE	0.259	0.741	456.077	0.001	0.221	60	60
	CAE	0.265	0.735	390.323	0.5	0.014	10	45
	CAE-AUG	0.330	0.65	315.349	0.5	0.087	40	29
	CAE-TL	0.309	0.691	407.289	0.75	0.001	50	50
Verbal Score (n=77)	**PCA**	0.318	0.682	362.837	1	0.034	40	40
	ICA	0.273	0.727	380.762	1	0.043	55	43
	Lin AE	0.246	0.754	296.648	0.5	0.152	30	23
	NonLin AE	0.262	0.738	368.885	0.75	0.014	40	40
	CAE	0.271	0.729	759.03	0.001	0.7	60	60
	CAE-AUG	0.398	0.607	315.57	1	0.0432	40	24
	CAE-TL	0.393	0.61	301.58	0.75	0.014	20	17

Table 4.1: Regression Metrics in the prediction of neuropsychological scores as a function of the feature extraction method obtained for the different feature extraction methods by applying ElasticNET with LOOCV. The value of the optimized parameters (λ, α, and k) and the number of non-zero features (NZ) are also reported. R^2: percentage of variance explained. MSE mean squared error, BIC Bayesian information criterion. Minimum MSE value, Minimum BIC value

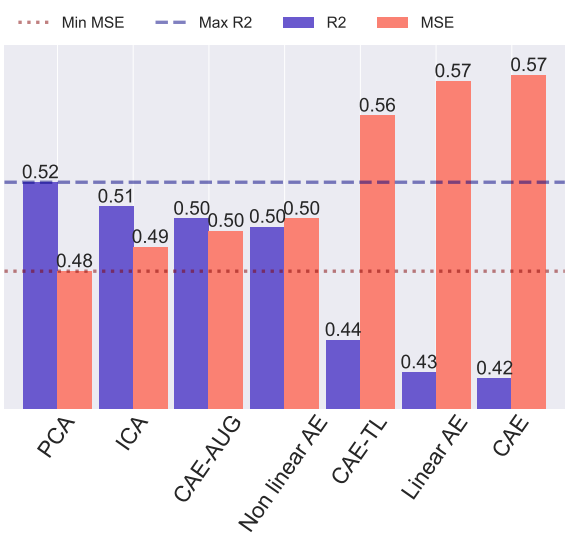
	Method	R^2	MSE	BIC	Optimal α	Optimal λ	Reg/k	NZ
Language Score	**CAE**-L_1	0.441	0.559	1120.703	1	0.002	0.1	200
	CAE-k	0.459	0.551	653.553	0.75	0.05	90	96
	TL-k	0.523	0.479	579.394	0.25	0.031	90	80
Spatial Score	**CAE**-k	0.280	0.720	490.7	0.75	0.034	60	60
	TL-k	0.310	0.690	237.755	1	0.115	30	7
Verbal Score	**CAE**-k	0.340	0.65	352.46	1	0.05	30	29
	TL-k	0.404	0.605	278.276	1	0.051	60	17

Table 4.2: Regression Metrics in the prediction of neuropsychological scores as a function of the feature extraction method obtained for the **overcomplete** version of the methods with latent space of size 200 by applying ElasticNET with LOOCV. The value of the optimized parameters (λ, α, and k) and the number of non-zero features (NZ) are also reported. R^2: percentage of variance explained. MSE mean squared error, BIC Bayesian information criterion.

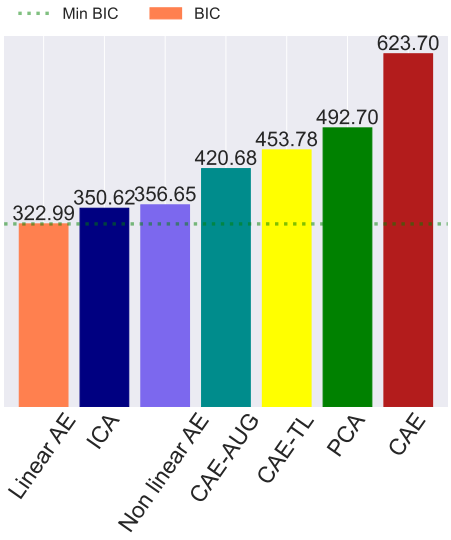
work. In general, it can be observed that the deep learning approach performs similar or worst than the conventional method in the language domain.

Figure 4.9b presents the BIC-metric obtained for the methods in the language domain. Interestingly, the autoencoder with asingle linear layer is often the one achieving the lowest BIC value, suggesting that such architecture is particularly useful to select a few representative components from the data distribution. Although PCA provides the lowest MSE metric, it gives a a high BIC-value. Moreover, the overcomplete - transfer learning model with k-sparsity as constraints leads to higher BIC value than PCA.

Regarding the Spatial Memory score, it can be observed from Figure 4.10a that the deep-learning methods for feature extraction performs better than the traditional ones (PCA and ICA). Moreover, PCA and ICA-based models are the ones leading to the worst results. In particular, the **CAE-AUG** was the one with the lowest MSE value. Similarly as in the case of language score, it can be observed that the $k-$sparsity constrain, in the overcomplete convolutional autoencoder, was leading to better performances than the undercomplete case



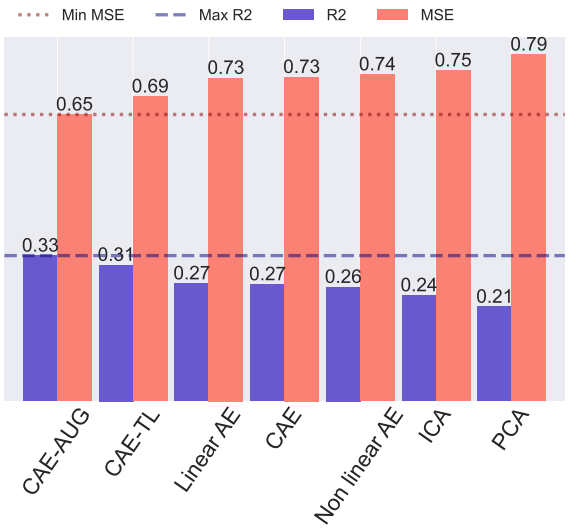
(a) MSE and R^2 sorted.



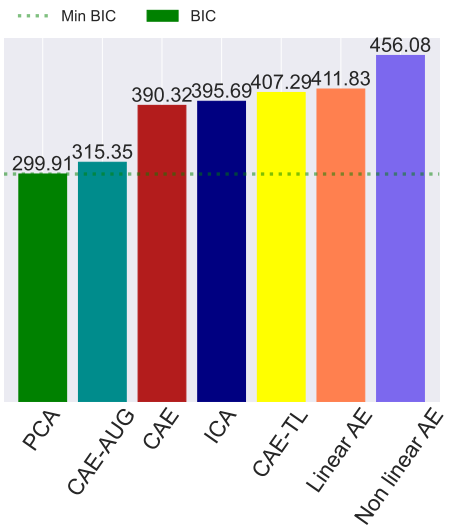
(b) BIC sorted

Figure 4.9: Metrics obtained using **language score** as neurophysiological value.

for transfer learning (TL) and the convolutional autoencoder (CAE) case. Once again the results obtained by the convolutional autoencoder directly applied to the stroke dataset did not perform correctly. Moreover, PCA-model and one-layer AE with linear and non linear activation function achieve similar metrics, which is no surprise since these two modes should perform similarly [33].Surprisingly, when observing the BIC-values in Figure 4.10b the overcomplete-k Transfer Convae and the PCA are the methods leading to the lowest value.



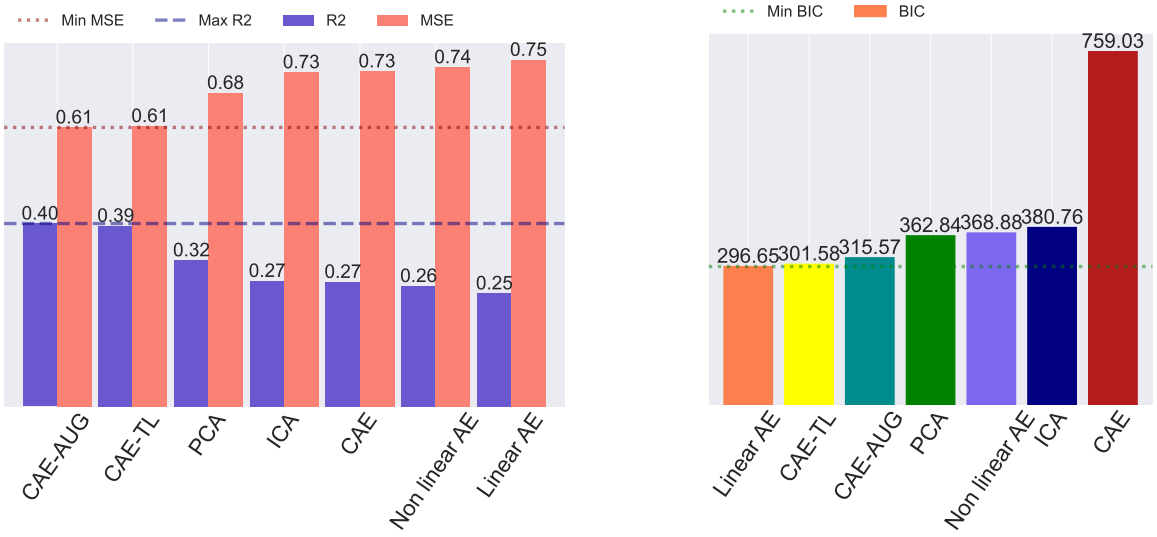
(a) MSE and R^2 sorted.



(b) BIC sorted

Figure 4.10: Metrics obtained using **spatial memory score** as neurophysiological value.

Finally, Figure 4.11a presents the MSE and R^2 values obtained for verbal memory score. Similarly as in the case of spatial memory, the deep-learning models were the ones leading to better accuracies, in fact, once again the augmentated were the one leading to the lowest MSE error, following by the transfer learning methods in the undercomplete and overcomplete case with k-sparcity constraint. The worst models are the simplest one (Linear AE, Non Linear AE, CAE). The lowest BIC values were also obtained from the DL-methods as it can be observed from Figure 4.11b. Similar as what happen with the spatial score, PCA provides better BIC-values than the ICA-model.



(a) MSE and R^2 sorted.

(b) BIC sorted

Figure 4.11: Metrics obtained using **verbal memory score** as neurophysiological value.

Figure 4.12 presents the optimal α and λ obtained for each model and neurophysiological score. Recall that λ controls the overall level (intesity) of regularization. As it can be observed, the λ parameter are usually small. On the other hand, it can be seen that the α value mainly takes the two extremas:

- $\alpha \sim 0$, which correspond to a Ridge regression;
- $\alpha \sim 1$, which correspond to a Lasso penalization;
- mixed-up $\alpha \sim 0.75$ (only happen in few cases).

Finally, Figure 4.13 presents the best total components in the fold and the number of non-zero features in them. It can be observed that the one layer convolutional autoencoder with linear and non linear activation function are the ones with lowest number of components in the language domain. Notice that this are also the ones providing the lowest BIC value. In the spatial domain, a similar behavior happens with PCA, whereas in the verbal memory domain it happen with the one layer convolutional autoencoder with linear activation function.

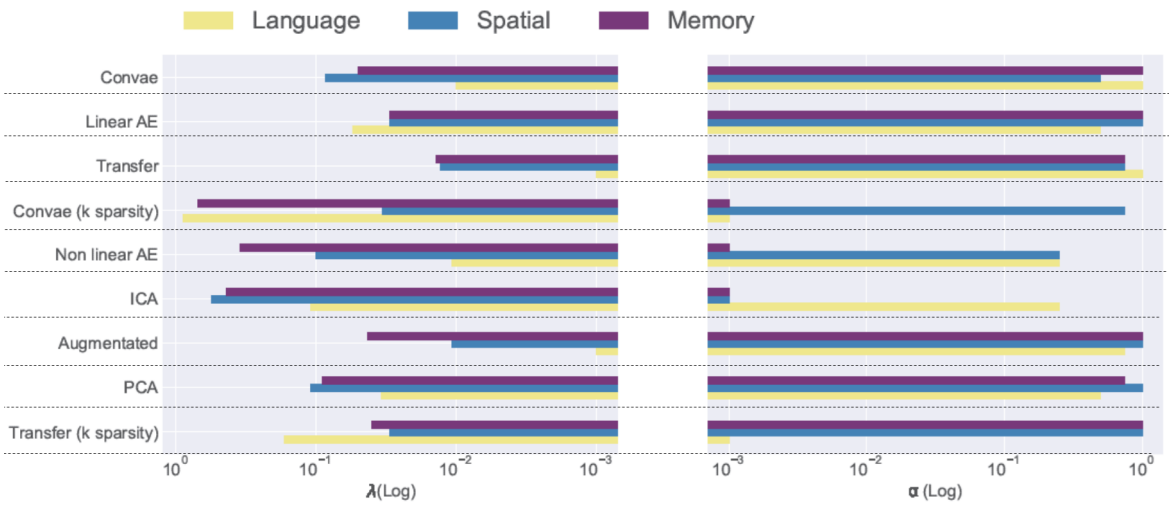
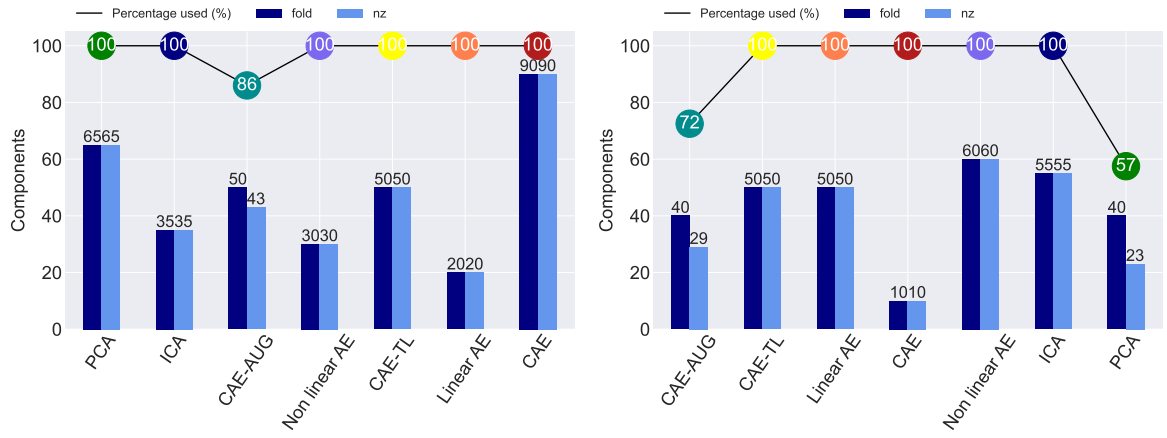
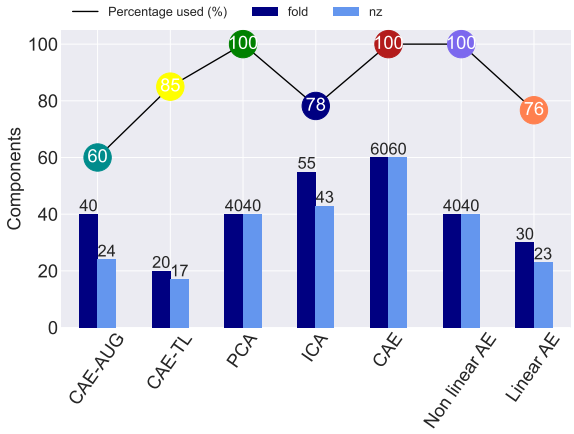


Figure 4.12: α and λ Elastic Net parameters obtained for the different models.



(a) Language score.

(b) Spatial memory score.



(c) Verbal memory score.

Figure 4.13: Fold used and number of non-zero features (nz). The circles represents the percentage of each of the folders used (i.e. $nz/Fold$).

The results showed that the deep learning models have similar performances in general than the traditional methods (ICA and PCA) on the resting-state fMRI dataset in the Language domain. However, this is not the case in the Spatial memory and Verbal memory domain, in which these models are better. In particular, the CAE-AUG model was the one leading to the best results when using the Spatial and Verbal memory as scores, with a considerable margin over the other methods. Such remarkable performance is approached also by the CAE trained using Transfer Learning.

4.2.1 *Getting deeper on Augmentation techniques*

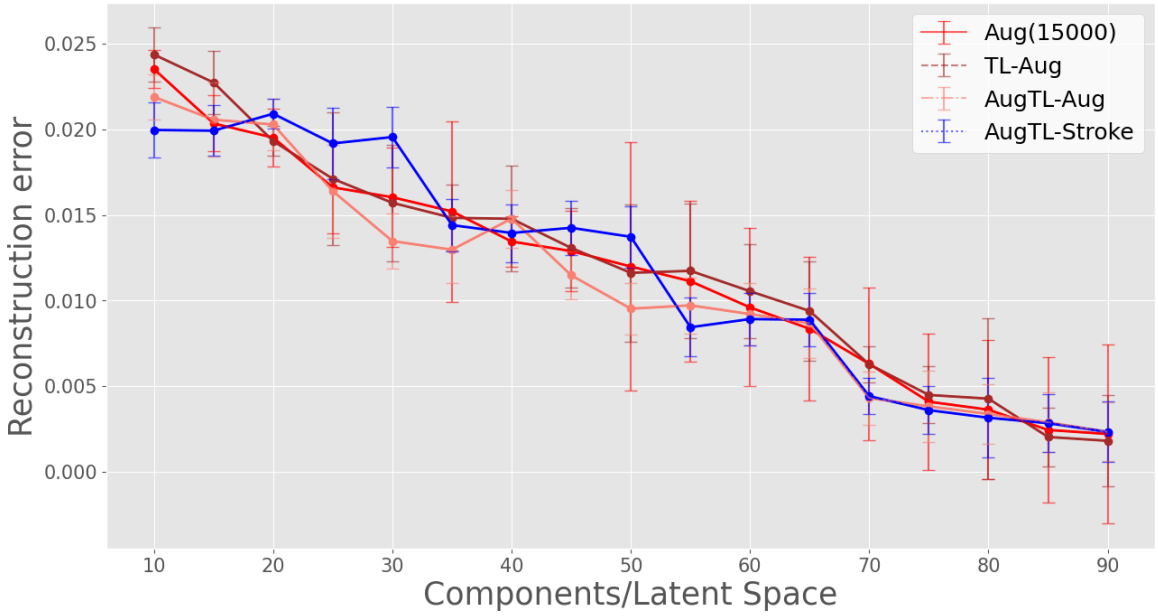


Figure 4.14: Reconstruction error for each augmentated dimensionality reduction method as a function of the number of extracted features.

As it was observed from Table 4.1, the augmentation techniques provides an improvement in the metrics due to the extension on the dataset. Figure 4.14 presents the reconstruction error obtained for the several augmentated methods. The errors are comparable to that achieved previously by the simpler Data Augmentation technique (Figure 4.1), suggesting that also in these cases we achieve very good reconstructions. Samples of reconstructed images for these methods are shown in Figures 4.15, 4.16, 4.17 and 4.18.

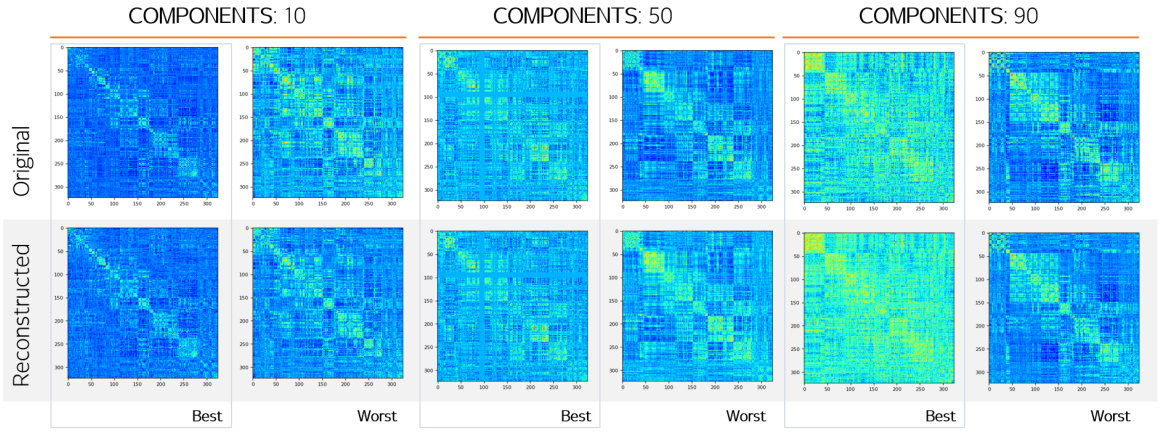


Figure 4.15: Best and worst reconstructed samples obtained by Aug(15000) with latent space 10, 50, 90.

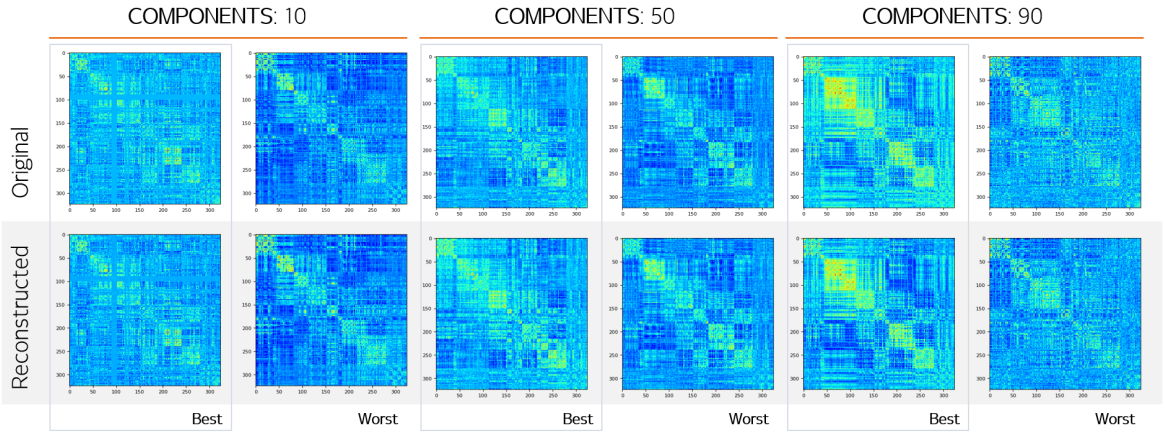


Figure 4.16: Best and worst reconstructed samples obtained by TL-Aug with latent space 10, 50, 90.

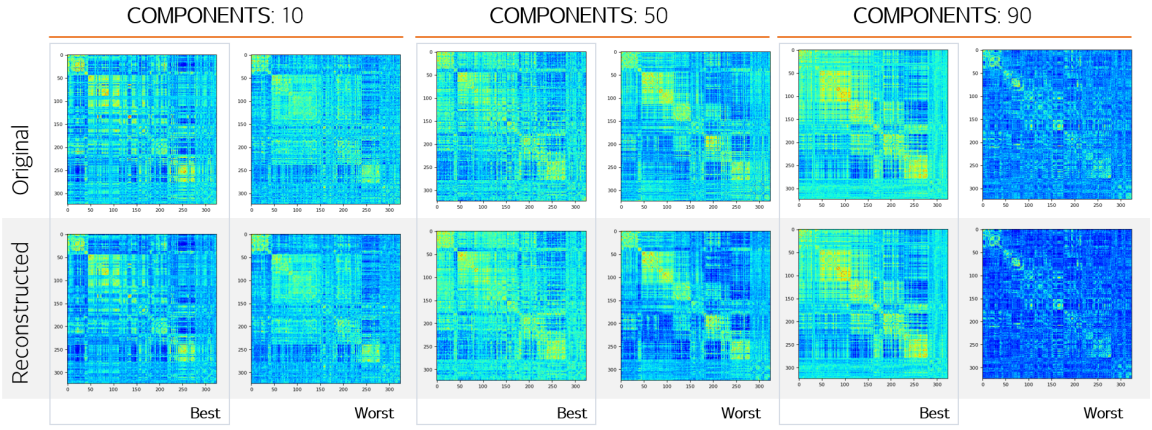


Figure 4.17: Best and worst reconstructed samples obtained by AugTL-Aug with latent space 10, 50, 90.

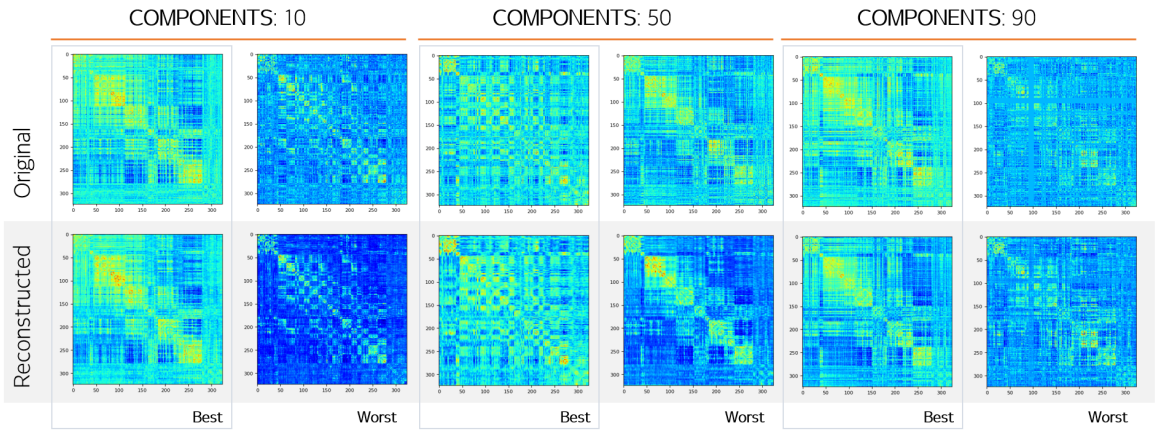


Figure 4.18: Best and worst reconstructed samples obtained by AugTL-Stroke with latent space 10, 50, 90.

However, one should notice that indeed the dataset is now more correlated than before, in fact, the similarity among image i and j was calculated as:

$$\text{similarity}[i,j] = 1 - \text{spatial.distance.cosine}(i,j) \tag{4.1}$$

where $\text{spatial.distance.cosine}(i,j)$ stands for the function provided by SCIPY that represents the cosine distance between u and v and is defined as:

$$1 - \frac{u \cdot v}{\|u\|_2 \|v\|_2}.$$

The mean value of the similarity matrix obtained by applying Equation 4.1 leads to an increase of $\sim 20\% - 31\%$ of the correlation among the images. However, the training loss and the validation loss are still similar, meaning that no underfit is observed on the data. Notice that the only set that was augmentated was the training set, whereas the validation and test set were remain the same with no changes.

Table 4.3 presents the results of the prediction metrics obtained by using as regressors the features for the augmentated techniques. Figure 4.19a presents the MSE and R^2 values obtained for the several methods in the language domain. Recall that, the best models obtained in Table 4.1 were the ones corresponding to the overcomplete transfer with k sparcity and the PCA-based model with a MSE equal to 0.48. As it can be observed, the transfer learning-model applied to augmentation stroke dataset was the one providing a dropped of $\sim 7\%$ of the MSE and a increase of the same amoung for the BIC-value. Following up, the transfer learning technique trained initially with the augmentated HCP dataset and then transfer to the augmentated stroke dataset also exhibits a lower MSE.

Figure 4.19b shows the prediction metrics obtained by the several augmentation techniques in the spatial memory domain. As in the previous domain 4.10a the augmentation techniques provides better accuracy than the traditional methods. In fact, the transfer

	Method	R^2	MSE	BIC	Optimal α	Optimal λ	Fold	NZ
Language Score	**TL- Aug**	0.555	0.445	283.634	0.001	0.031	20	20
	Aug(15000)	0.514	0.486	420.68	0.5	0.06	50	43
	AugTL-Stroke	0.468	0.532	432.587	1	0.016	60	46
	AugTL-Aug	0.534	0.456	420.68	0.5	0.06	50	43
Spatial Score	**TL-Aug**	0.395	0.565	367.205	0.5	0.098	70	42
	Aug(15000)	0.359	0.581	569.703	0.001	0.05	15	85
	AugTL-Stroke	0.282	0.718	380.29	0.001	0.811	40	40
	AugTL-Aug	0.234	0.766	246.493	1	0.159	55	9
Verbal Score	**TL-Aug**	0.469	0.541	357.279	0.75	0.083	70	37
	Aug(15000)	0.410	0.589	569.703	0.001	0.05	15	85
	AugTL-Stroke	0.420	0.580	242.202	1	0.159	45	8
	AugTL-Aug	0.427	0.571	238.96	1	0.083	25	8

Table 4.3: Regression Metrics in the prediction of neuropsychological scores as a function of the feature extraction method obtained for the augmentated models by applying ElasticNET with LOOCV. The value of the optimized parameters (λ, α, and k) and the number of non-zero features (NZ) are also reported. R^2: percentage of variance explained. MSE mean squared error, BIC Bayesian information criterion

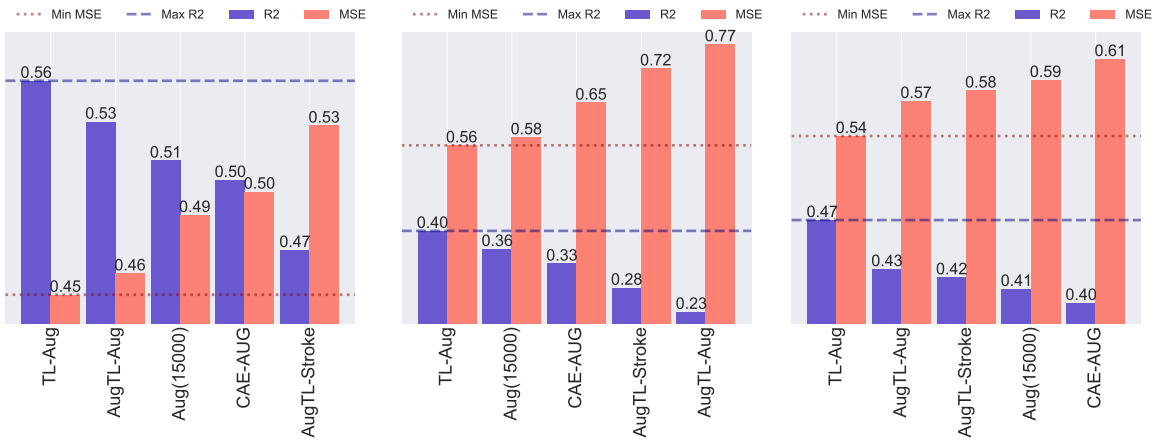
learning-model applied to augmentation stroke dataset was the one with the lowest MSE-score, dropping the value about $\sim 30\%$ with respect to the PCA-model, and $\sim 15\%$ with respect to the ICA-model. Moreover, the R^2 value increased significantly ($\sim 66\%$) with respect to the PCA and ICA-models.

Finally, Figure 4.19c shows the prediction metrics obtained by the several augmentation techniques in the verbal memory domain. In this domain it can be observed that increasing the size of the dataset provides better accuracy, than all the remain methods. In particular, as before, the transfer learning-model applied to augmentation stroke dataset was the one

with the lowest MSE-score. In fact, with respect to the PCA-based model the MSE value dropped $\sim 20\%$, and $\sim 27\%$ with respect to the ICA-based model. Furthermore, the R^2 value gained about 47% with respect to the PCA and ICA-models.

The larger accuracy gains for memory scores can be explained by the fact that prediction of language scores is likely close to ceiling. Memory has a more distributed neural basis and the prediction of deficits from structural lesions is relatively poor compared to other behavioral domains [50, 51]. Therefore, predicting memory scores represents an important benchmark for RSFC-based machine learning methods.

In general, Regression results reported in Table 4.3 clearly show that these improved data augmentation and transfer learning regimens further boosted the model's performance, both in terms of MSE and R^2. All regimens generally enhance the CAE accuracy, however the most striking improvement is given by the TL-Aug regimen, which reaches significantly better performance compared to all methods previously investigated, establishing a new state-of-the-art for the stroke-prediction task. Interestingly, this improved model achieves such accurate predictions by relying, on average, on fewer components compared to other methods, which might be particularly relevant to improve interpretability of the resulting model.



(a) Language domain (b) Spatial memory (c) Verbal memory

Figure 4.19: MSE and R^2 sorted in the all domain for the augmentated cases

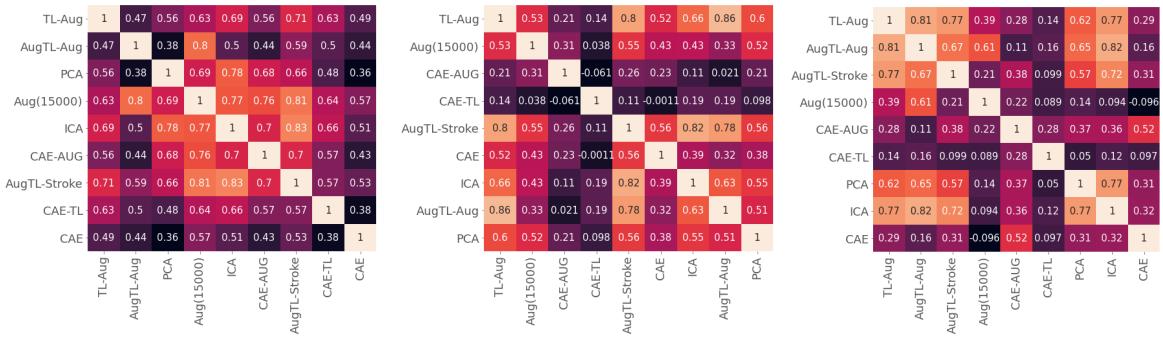
Additionally, in the language score it can be observed from Table 4.3 that the TL-Aug model not only provies the lowest MSE but also the lowest BIC value in the language domain. Moreover, the AugTL-Aug gets the lowest BIC-value in the spatial memory and the verbal memory domains, suggesting that such architecture is particularly useful to select a few representative components from the data distribution. Once again, this values corresponds to the lowest NZ for each model.

Finally, the α-values obtained in Table 4.3, shows something similar as before, either α is one or zero (purely Ridge or purely Lasso), or some intermediate value.

4.3 MAPS OF PREDICTIVE FUNCTIONAL CONNECTIVITY EDGES

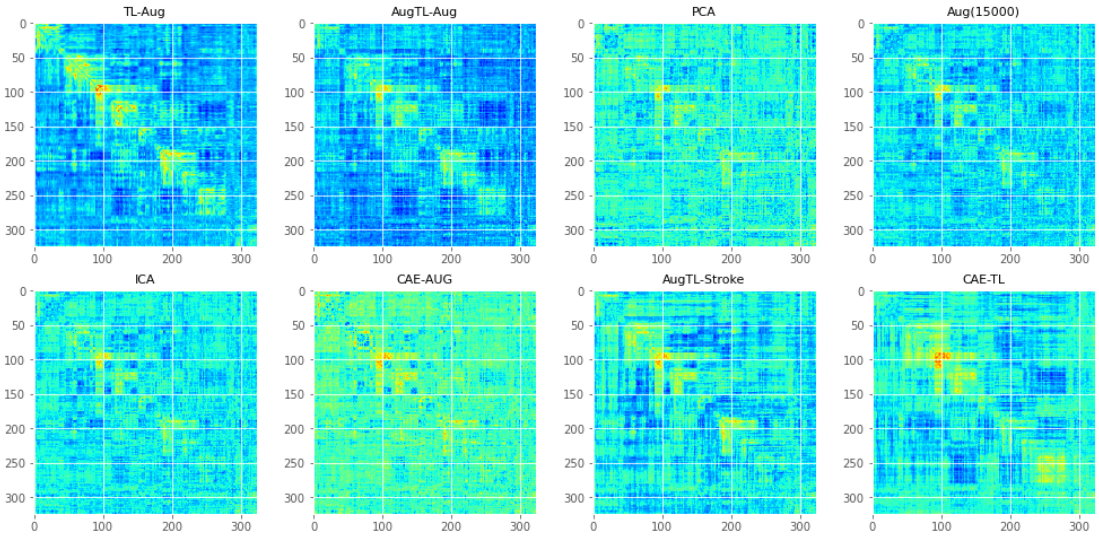
Figure 4.21 presents the backpropagation of the optimal regression coefficients for all domains. In order to have a better comparison, Figure 4.20 shows the confusion matrix obtained by computing cosine similarity between the images obtained sorted by least MSE. First of all, it should be notice that the back-projection in the language and the verbal memory domain of PCA is highly correlated to the ICA model. In the language domain, all the models are quite similar with ranges going from $\sim [0.4 - 0.9]$. Additionally, CAE-AUG and AUG(15000) were similar to the ICA-based model.

In the Spatial domain, it can be observed that the backprojective maps obtained from the CAE-TL and the CAE-AUG are the lest similar ones, providing negative values. The most similar predictive maps are between TL-Aug and AugTL-Aug. The same behavior can be observed in the memory domain. In fact, once again the CAE-TL maps are the least similar among the others, and this is evident when observing Figure 4.21.

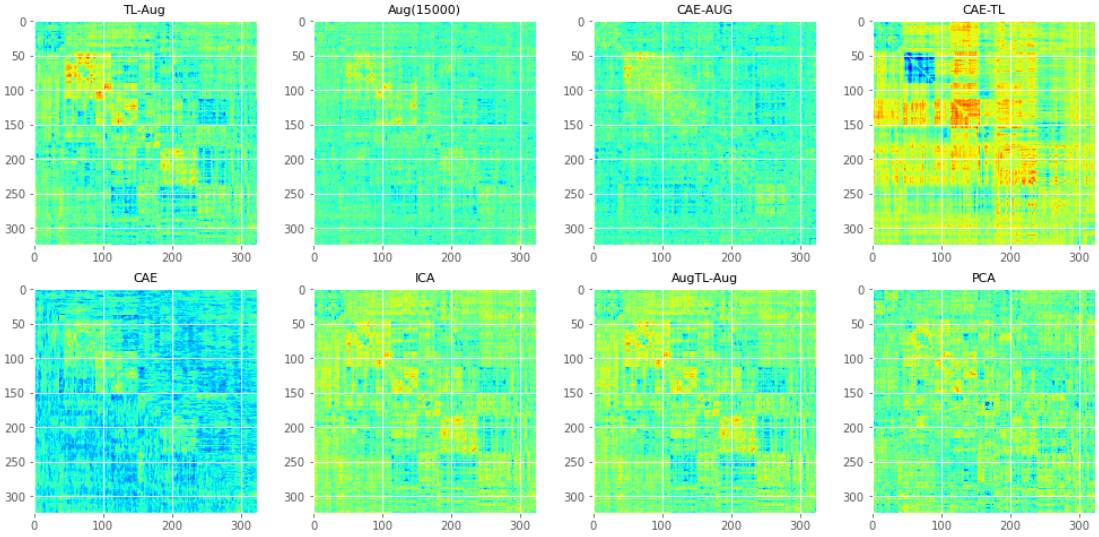


(a) Language score. (b) Spatial memory score. (c) Verbal memory score.

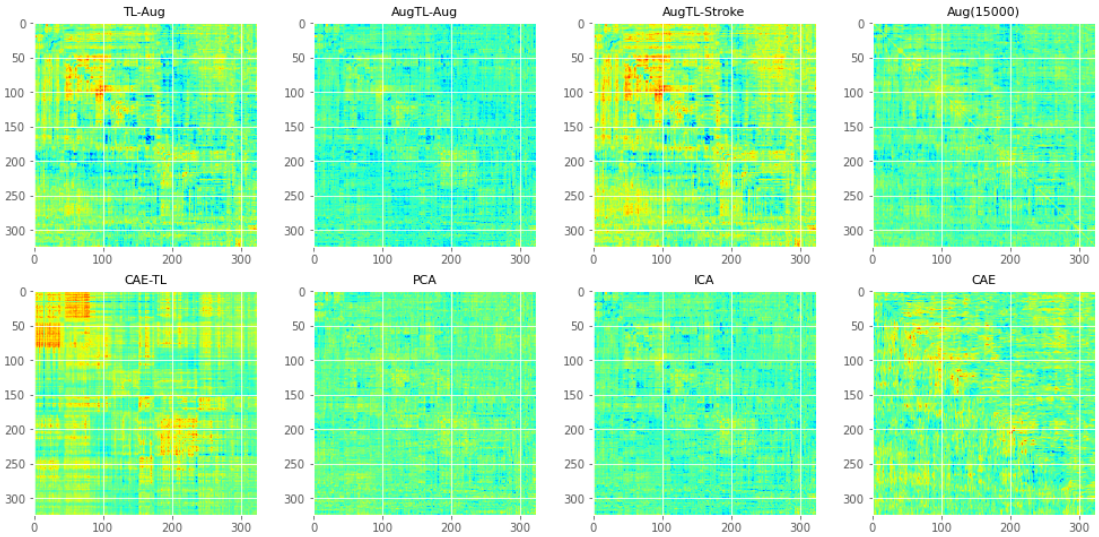
Figure 4.20: Confusion matrix obtained by computing the similarity among each predictive map obtained by back-projecting the regression coefficients.



(a) Language score.



(b) Spatial memory score.



(c) Verbal memory score.

Figure 4.21: Maps of predictive functional connectivity edges obtained by back-projecting the regression coefficients.

4.4 CROSS-VALIDATION SETUP AND MODEL ESTIMATION

Figures 4.22 and 4.23 presents a comparison of the MSE and BIC value obtained by a a standard cross-validation scheme (LOOCV) with a nested cross-validation approach (NLOOCV) in the spatial domain. In both cases, it can be observed that the n-mode condition is leading to the same values as in the case of LOOCV. A slightly variance in the mean value obtained in the n-mean and n-median case can be found in contrast to the LOOCV-scheme. In particular this is more notorious when observing the BIC-values, specifically when dealing with the overcomplete convolutional autoencoder with k-sparsity and the transfer model. However, this results are consistent with the ones obtained by Calesella, Testolin, De Filippo De Grazia, and Zorzi [9], in which they compared the R^2 and BIC values in the language domain for the same dataset. This disparity of the n-mean and n-median was explained due to the high susceptibility of the mean to outliers, so that major departures from the distribution of the selected parameters could drive the mean toward the outlier values [9].

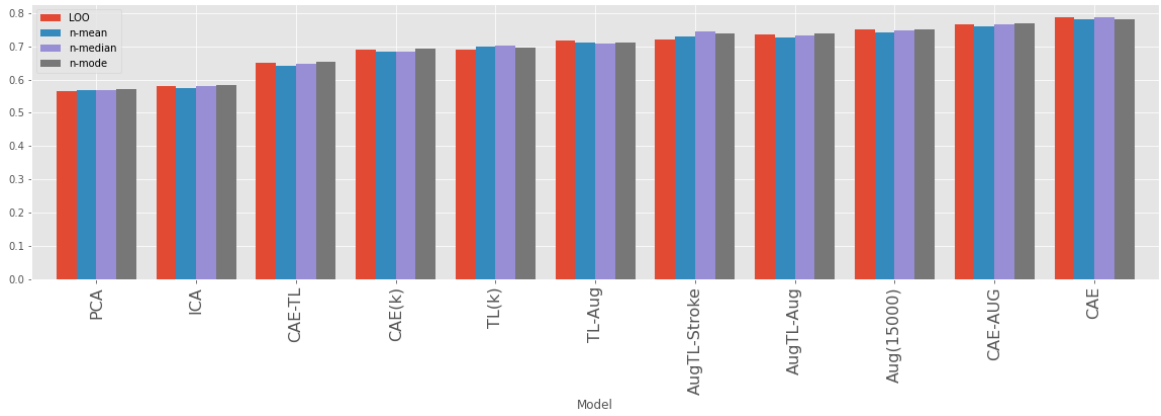


Figure 4.22: MSE differences across the CV schemes for each feature extraction method

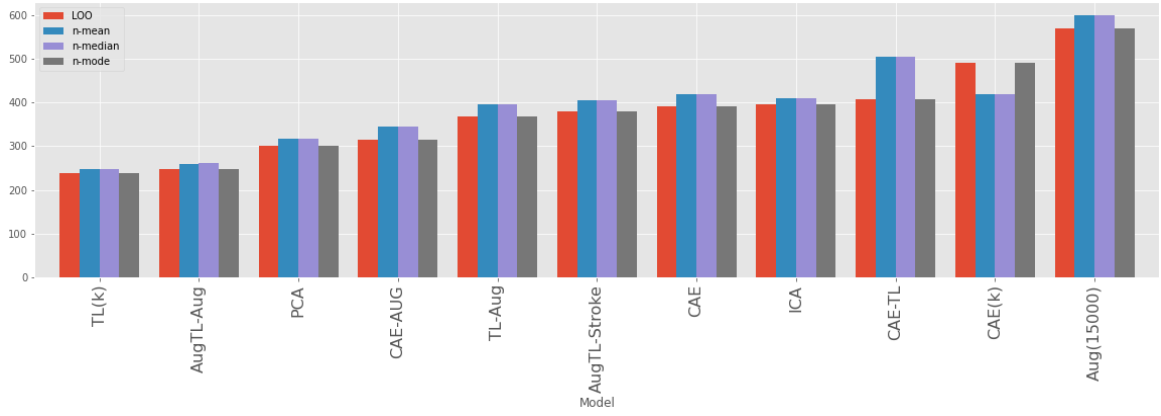


Figure 4.23: BIC differences across the CV schemes for each feature extraction method in the spatial domain.

To sum up, in the spatial memory domain, the effect of the nested CV scheme upon model performance is the same one obtained as in the LOOCV-scheme when working with the n-mode condition. However, the n-mean and n-median leads to higher/lower values same as Calesella, Testolin, De Filippo De Grazia, and Zorzi [9] but with negligible difference. In conclusion, the LOOCV setup allows us to obtain an optimal selection of the parameters of the model with the main advantage of time-complexity.

5

CONCLUSIONS

In this work we investigated whether deep autoencoders could extract relevant features from resting state functional connectivity data of stroke patients, which can successively be used to build predictive models of neuropsychological scores. We implemented a variety of autoencoder architectures, ranging from simple, one-layer linear networks to more sophisticated convolutional versions exploiting several layers of non-linear processing. In order to deal with the issue of data scarcity, which is known to affect the performance of deep learning models, we also explored data augmentation and transfer learning techniques. The autoencoder's performance was benchmarked against other conventional approaches, such as Principal Componenty Analysis (PCA) and Independent Component Analysis (ICA).

The different methods were first evaluated in terms of their reconstruction error. In general, all methods achieved similar reconstruction error, though the autoencoders trained using data augmentation obtained slightly better accuracy. The quality of the features extracted by different methods was then assessed based on their capacity to serve as predictors for neuropsychological scores of the patients in three cognitive domains (i.e., language, spatial memory, and verbal memory). To this aim, the extracted features were given as input to regularized regression models, and performance was evaluated in terms of coefficient of determination, mean squared error and Bayesian information criterion. Results showed that the performance of the basic autoencoders was overall comparable to that of traditional methods (ICA and PCA). However, more sophisticated convolutional architectures trained using data augmentation and transfer learning achieved a much higher performance, with considerable gains of 7% (language), 66% (spatial memory) and 47% (verbal memory) with respect to the previously reported state-of-the-art methods [9].

In conclusion, our results demonstrate the great potential of deep learning models for the analysis of multi-dimensional neuroimaging data even in cases with limited data availability, which is often considered a critical limitation in clinical studies. Future work should aim at further consolidating our findings, for example by systematically evaluating the performance of deep learning models on the prediction of other neuropsychological and behavioral scores. Moreover, a key research frontier would be to design and implement advanced visualization techniques in order to interpret the features extracted by non-linear dimensionality reduction

methods, which could provide valuable insights to the clinicians for the design of more effective rehabilitation protocols.

A

APPENDIX

In this section, we will present the hyperparameters obtained for the autoencoders in order to obtain better performances

A.1 HYPERPARAMETER TUNING

Hyperparameter optimization simply consist on searching the best set of hyperparameters that gives the best version of a model on a particular dataset. The optimal hyperparamenters obtained for each model by means of OPTUNA [44] are present in the next subsections. It should be point out that this plays a key role in achieving better performances of the model.

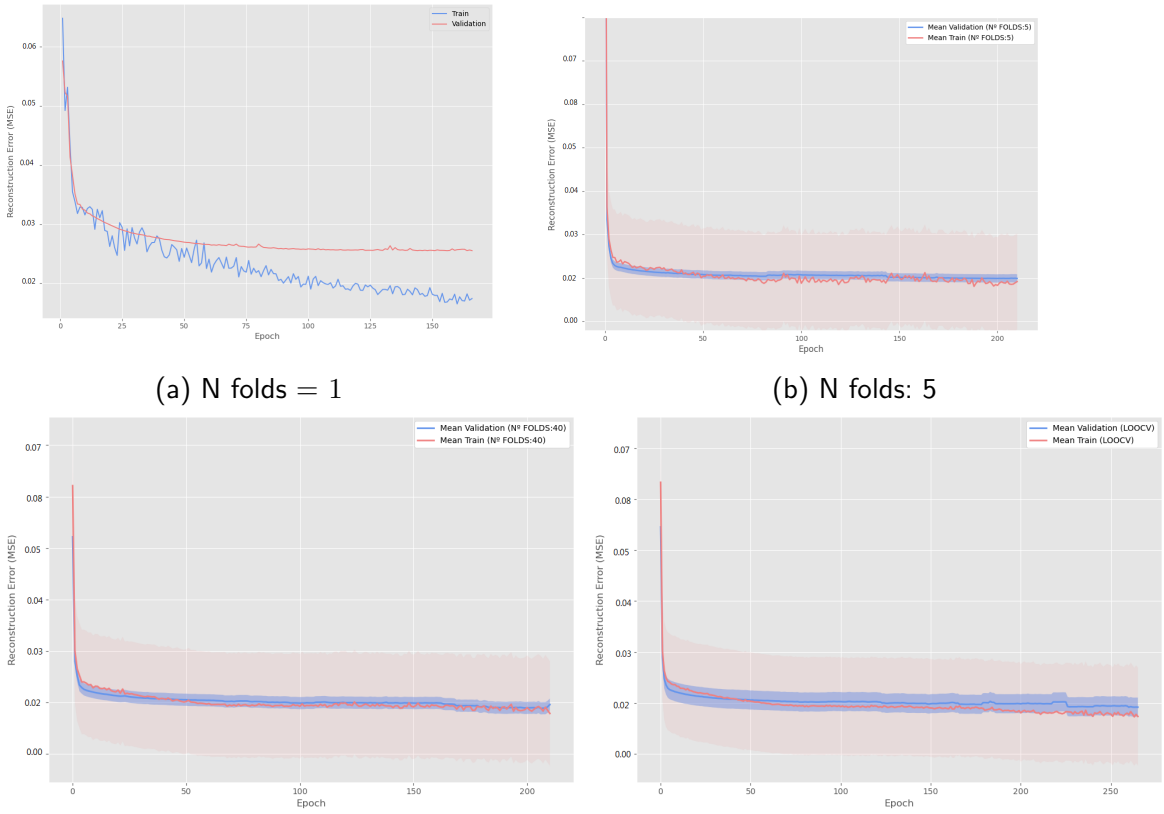
A.1.1 *CAE*

When working with deep learning models, in order to achieve good performances, large dataset are needed. As already mentioned, the stroke dataset consist of only 132 patients, therefore, the impact of applying $K - fold$ cross validation is study in terms of performances and time. After assessing the number of folds to used, the best hyperparameters are presented.

Figure A.1a presents the learning curve for the convolutional autoencoder without using cross validation. As it can be observed, after the 50 epoch, the validation loss remains in a plateau without improving its value whereas the train loss still decrease. This behaviour is refer as overfitting. Large dataset helps us avoid overfitting and generalizes better as it captures the inherent data distribution more effectively, however, the stroke dataset available is small. Therefore, cross validation is used to tune the hyperparameters in order to avoid this behaviour. The major advantage of any form of cross-validation is that each result is generated using a classifier which was not trained on that result.

In order to determine the number of folds to use, the performances of `None`, 5, 40 and 119 (*-LOOCV-*) folds are compared. Figures A.1b, A.1c and A.1d presents the learning curves for the model when using cross validation by means KFOLD with $K = 5, 40, 119$ respectively. It can be observed that the mean value of the validation decreases as well as

the mean value of the training loss when only using 5 folds. In general a similar behavior of the learning curves can be observed when working with the different folds.



(a) N folds $= 1$

(b) N folds: 5

(c) N folds: 40

(d) N folds: 119

Figure A.1: Learning curves for the convolutional autoencoder.

Additionally, in order to asses the optimal number of folders, the time complexity on performing the Kfold cross validation are considered. Table A.1 presents the time complexity for each number of folds. As expected, the larger the number of folder, the more iterations to performed and therefore the more times it takes. Moreover, 50 trials are performed for each model to find the optimal hyperparameters for the several latent space $[10, 90]$ in steps of 5. Therefore, in order to get a good generalization of the data, without increasing too much the time complexity, 5 folds are used to tune the hyperparameters in the model since, as it was already shown in Figure A.1, five folds were enough in order to get a good generalization of the problem and overcome overfitting issues as shown in Figure A.1a.

Nº folds	0	5	40	119
Time [min]	~ 4	~ 18	~ 236	~ 5700

Table A.1: Time complexity against number of folds.

Table A.2 present the optimal hyperparameters obtained for the best latent space obtained for each of the regression metrics[1], after 50 trials for the convolutional autoencoder model present in Figure 3.4 using 5-KFOLD cross validation. As it can be observed, `Adam` was the best optimizer to used in all cases, therefore, in the following sections we will restrict ourselves only to that one. Dropout and weight decay are two forms of regularization. As it can be observed from Table A.2, dropout plays an important role in this model with non-zeros values allowing generalization. However, `weight decay` values are usually quite small ($\sim 1e^{-5} - 1e^{-5}$) and it plays a relevant role when SGD optimizer is used.

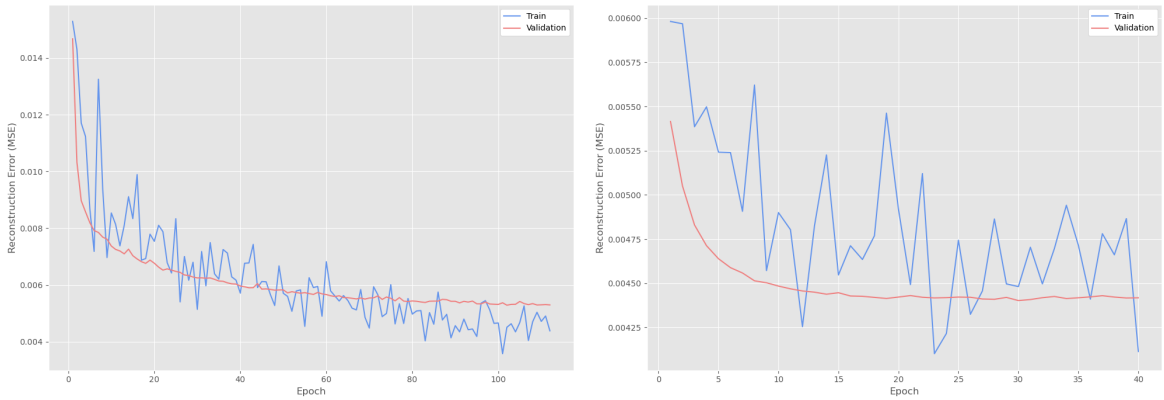
Table A.2: Optimal hyperparameter values for CAE-model found by minimizing the mean of the validation loss of 5-KFOLD by means of OPTUNA [44]

	Latent Space	Conv1	Conv2	Conv3	dropout	fc	lr	opt	weight
Language	90	128	128	16	0.36	64	0.002	Adam	0.00001
Spatial memory	45	128	32	16	0.091	128	0.0002	Adam	0.00002
Verbal memory	60	128	32	32	0.111	128	0.0005	Adam	0.00008

A.1.2 *CAE-TL*

Figure A.2a presents the learning curves of the convolutional autoencoder learnt by using the Human Connectome Project (HCP) dataset. As it can be observed, the validation and training losses converge to a similar value. Therefore, no overfitting is observed, as it was previously seen from Figure A.1a, and no cross validation is applied since the dataset is already quite exhaustive. On the other hand, Figure A.2b presents the validation and training losses obtained after applying transfer learning to the original dataset using the pre-trained convolutional autoencoder. In contrast of what happened with the convolutional autoencoder applied directly to the stroke dataset (figure A.1) the model converges faster (in earlier number of epochs) achieving similar performances.

1 Dropout values, learning rates and weight decays values were rounded in all models to proper display the tables



(a) (b)

Figure A.2: (a) Learning curves for the convolutional autoencoder with latent space equal to 90 trained with the Human Connectome Project dataset. (b) Learning curves obtained after applying transfer learning to the original stroke dataset from the HCP one with latent space equal to 90.

Table A.3 presents the hyperparameters obtained for the best latent space obtained for each of the regression metrics for the CAE-TL-model. Additionally, A.4 presents the optimal hyperparameters of the same model apply to the stroke dataset with frozen convolutional trainable part.

Table A.3: Optimal hyperparameter values for CAE-TL-model found by minimizing the validation loss by means of OPTUNA [44], using the HCP dataset

	Latent Space	Conv1	Conv2	Conv3	dropout	fc	lr	opt
Language	50	16	32	128	0.411	64	0.0001	'Adam'
Spatial memory	50	16	32	128	0.411	64	0.0001	'Adam'
Verbal memory	20	64	128	128	0.226	64	0.0004	'Adam'

Table A.4: Optimal hyperparameter values for CAE-TL-model found by minimizing the validation loss by means of OPTUNA [44], using the *stroke* dataset.

	Latent Space	fc	lr
Language	50	8	0.0009
Spatial memory	50	8	0.0009
Verbal memory	20	16	0.0002

A.1.3 *CAE-AUG*

Table A.5 presents the hyperparameters obtained for the best latent space obtained for each of the regression metrics for the CAE-AUG-model.

Table A.5: Optimal hyperparameter values for CAE-AUG-model found by minimizing the validation loss by means of OPTUNA [44], using the *stroke* dataset.

	Latent Space	Conv1	Conv2	Conv3	dropout	fc	lr	opt
Language	50	16	8	16	0.105	128	0.0014	'Adam'
Spatial memory	40	16	32	32	0.243	128	8.7e-05	'Adam'
Verbal memory	40	16	32	32	0.243	128	8.7e-05	'Adam'

A.1.4 *AUG (15000)*

Table A.6 presents the hyperparameters obtained for the best latent space obtained for each of the regression metrics for the AUG(15000)-model.

Table A.6: Optimal hyperparameter values for AUG(15000)-model found by minimizing the validation loss by means of OPTUNA [44], using the *stroke* dataset.

	Latent Space	Conv1	Conv2	Conv3	dropout	fc	lr	opt
Language	50	16	8	128	0.297	8	0.006	'Adam'
Spatial memory	15	32	64	32	0.48	32	0.0005	'Adam'
Verbal memory	15	32	64	32	0.48	32	0.0005	'Adam'

A.1.5 *Aug-Stroke*

Table A.7 presents the hyperparameters obtained for the best latent space obtained for each of the regression metrics for the Aug-stroke-model apply on the HCP dataset. Additionally, A.8 presents the optimal hyperparameters of the same model apply to the stroke dataset with frozen convolutional trainable part.

Table A.7: Optimal hyperparameter values for Aug-Stroke-model found by minimizing the validation loss by means of OPTUNA [44], using the HCP dataset.

	Latent Space	Conv1	Conv2	Conv3	dropout	fc	lr	opt
Language	60	16	32	32	0.59	'fc	64	1.76e-05 'Adam'
Spatial memory	40	16	64	64	0.501	32	0.003	'Adam'
Verbal memory	45	8	128	16	0.59	64	0.001	'Adam'

Table A.8: Optimal hyperparameter values for Aug-Stroke-model found by minimizing the validation loss by means of OPTUNA [44], using the *stroke* dataset.

	Latent Space	fc	lr
Language	60	32	0.0002
Spatial memory	40	16	0.005
Verbal memory	45	64	0.005

A.1.6 *Aug-Aug*

Table A.9 presents the hyperparameters obtained for the best latent space obtained for each of the regression metrics for the Aug-Aug-model apply on the HCP dataset. Additionally, A.10 presents the optimal hyperparameters of the same model apply to the stroke dataset with frozen convolutional trainable part.

Table A.9: Optimal hyperparameter values for Aug-Stroke-model found by minimizing the validation loss by means of OPTUNA [44], using the HCP dataset.

	Latent Space	Conv1	Conv2	Conv3	dropout	fc	lr	opt
Language	50	64	128	128	0.44	16	0.001	'Adam'
Spatial memory	55	64	8	64	0.16	64	0.001	'Adam'
Verbal memory	25	8	16	128	0.103	32	6.58e-05	'Adam'

Table A.10: Optimal hyperparameter values for Aug-Stroke-model found by minimizing the validation loss by means of OPTUNA [44], using the $stroke$ dataset.

	Latent Space	fc	lr
Language	50	128	0.0042
Spatial memory	55	128	0.00015
Verbal memory	25	64	0.0006

BIBLIOGRAPHY

[1] S. Sarraf and G. Tofighi, "Classification of alzheimer's disease using fmri data and deep learning convolutional neural networks", Mar. 2016.

[2] M. Greicius, K. Supekar, V. Menon, and R. Dougherty, "Resting-state functional connectivity reflects structural connectivity in the default mode network", *Cereb Cortex*, vol. 19, pp. 72–8, Dec. 2008. DOI: 10.1093/cercor/bhn059.

[3] D.-E. Meskaldji, M. G. Preti, T. Bolton, M.-L Montandon, C. Rodriguez, S. Morgenthaler, P. Giannakopoulos, S. Haller, and D. Van De Ville, "Prediction of long-term memory scores in mci based on resting-state fmri", *NeuroImage: Clinical*, vol. Volume 12, 785–795, Oct. 2016. DOI: 10.1016/j.nicl.2016.10.004.

[4] M. Thomason, E. Dennis, A. Joshi, *et al.*, "Resting-state fmri can reliably map neural networks in children", *NeuroImage*, vol. 55, pp. 165–75, Mar. 2011. DOI: 10.1016/j.neuroimage.2010.11.080.

[5] J. S. Siegel, L. E. Ramsey, A. Z. Snyder, N. V. Metcalf, R. V. Chacko, K. Weinberger, A. Baldassarre, C. D. Hacker, G. L. Shulman, and M. Corbetta, "Disruptions of network connectivity predict impairment in multiple behavioral domains after stroke", *Proceedings of the National Academy of Sciences*, vol. 113, no. 30, E4367–E4376, 2016.

[6] J.-H. Kim, Y. Zhang, K. Han, M. Choi, and Z. Liu, "Representation learning of resting state fmri with variational autoencoder", Jun. 2020. DOI: 10.1101/2020.06.16.155937.

[7] H. Yamaguchi, Y. Hashimoto, G. Sugihara, J. Miyata, T. Murai, H. Takahashi, M. Honda, A. Hishimoto, and Y. Yamashita, "Three-dimensional convolutional autoencoder extracts features of structural brain images with a diagnostic label-free approach: Application to schizophrenia datasets", Aug. 2020. DOI: 10.1101/2020.08.24.213447.

[8] M. Khosla, K. Jamison, G. Ngo, A. Kuceyeski, and M. Sabuncu, "Machine learning in resting-state fmri analysis", *Magnetic Resonance Imaging*, vol. 64, Jun. 2019. DOI: 10.1016/j.mri.2019.05.031.

[9] F. Calesella, A. Testolin, M. De Filippo De Grazia, and M. Zorzi, "A comparison of feature extraction methods for prediction of neuropsychological scores from functional connectivity data of stroke patients", *Brain Informatics*, vol. 8, Dec. 2021. DOI: 10.1186/s40708-021-00129-1.

[10] L. Jollans, R. Boyle, E. Artiges, *et al.*, "Quantifying performance of machine learning methods for neuroimaging data", *NeuroImage*, vol. 199, Jun. 2019. DOI: 10.1016/j.neuroimage.2019.05.082.

[11] S. Wold, K. H. Esbensen, and P. Geladi, "Principal component analysis", *Chemometrics and Intelligent Laboratory Systems*, vol. 2, pp. 37–52, 1987.

[12] B. Cai, G. Zhang, A. Zhang, L. Xiao, W. hu, J. Stephen, T. Wilson, V. Calhoun, and Y. Wang, "Functional connectome fingerprinting: Identifying individuals and predicting cognitive functions via autoencoder", *Human Brain Mapping*, vol. 42, Apr. 2021. DOI: 10.1002/hbm.25394.

[13] A. Sólon, A. Franco, C. Craddock, A. Buchweitz, and F. Meneguzzi, "Identification of autism spectrum disorder using deep learning and the abide dataset", *NeuroImage: Clinical*, vol. 17, Aug. 2017. DOI: 10.1016/j.nicl.2017.08.017.

[14] J. Kim, V. Calhoun, E. Shim, and J.-H. Lee, "Deep neural network with weight sparsity control and pre-training extracts hierarchical features and enhances classification performance: Evidence from whole-brain resting-state functional connectivity patterns of schizophrenia", *NeuroImage*, vol. 124, May 2015. DOI: 10.1016/j.neuroimage.2015.05.018.

[15] W. Pinaya, A. Mechelli, and J. Sato, "Using deep autoencoders to identify abnormal brain structural patterns in neuropsychiatric disorders: A large-scale multi-sample study", *Human Brain Mapping*, vol. 40, Oct. 2018. DOI: 10.1002/hbm.24423.

[16] H. Huang, X. Hu, Y. Zhao, M. Makkie, Q. Dong, S. Zhao, K. Li, and T. Liu, "Modeling task fmri data via deep convolutional autoencoder", *IEEE Transactions on Medical Imaging*, vol. PP, pp. 1–1, Jun. 2017. DOI: 10.1109/TMI.2017.2715285.

[17] X.-F. GENG and J. Xu, "Application of autoencoder in depression diagnosis", *DEStech Transactions on Computer Science and Engineering*, Dec. 2017. DOI: 10.12783/dtcse/csma2017/17335.

[18] D. Bank, N. Koenigstein, and R. Giryes, "Autoencoders", Mar. 2020.

[19] D. C. Van Essen, S. M. Smith, D. M. Barch, T. E. Behrens, E. Yacoub, K. Ugurbil, W.-M. H. Consortium, *et al.*, "The wu-minn human connectome project: An overview", *Neuroimage*, vol. 80, pp. 62–79, 2013.

[20] Z. Cui and G. Gong, "The effect of machine learning regression algorithms and sample size on individualized behavioral prediction with functional connectivity features", *NeuroImage*, vol. 178, Jun. 2018. DOI: 10.1016/j.neuroimage.2018.06.001.

[21] J. Arias Almeida, P. Ciuciu, M. Dojat, F. Forbes, A. Frau-Pascual, T. Perret, and J. Warnking, "Pyhrf: A python library for the analysis of fmri data based on local estimation of the hemodynamic response function", Jul. 2017. DOI: 10.25080/shinma-7f4c6e7-006.

[22] I. Tsougos, *Advanced MR Neuroimaging: From Theory to Clinical Practice*, 1st ed., ser. Series in Medical Physics and Biomedical Engineering. CRC Press, 2018, ISBN: 1498755232; 9781498755238; 9781351216524; 135121652X; 9781351216531; 1351216538; 9781351216548; 1351216546. [Online]. Available: libgen.li/file.php?md5=c58e5e17c764392a3ab5a2c88f571031.

[23] C. F. B. Janine Bijsterbosch Stephen M. Smith, *An Introduction to Resting State fMRI Functional Connectivity*, 1st ed., ser. Oxford Neuroimaging Primers. OXFORD UNIVERSITY PRESS, 2017, ISBN: 9780198808220; 0198808224. [Online]. Available: libgen.li/file.php?md5=4a073f77e44994b4b0761c2f48ae2c06.

[24] T. E. N. Russell A. Poldrack Jeanette A. Mumford, *Handbook of Functional MRI Data Analysis*, 1st ed. Cambridge University Press, 2011, ISBN: 0521517664; 9780521517669. [Online]. Available: libgen.li/file.php?md5=30fb01a51f981ff1d823e01145338776.

[25] I. Tsougos, *Advanced MR Neuroimaging: From Theory to Clinical Practice*, 1st ed., ser. Series in Medical Physics and Biomedical Engineering. CRC Press, 2018, ISBN: 1498755232; 9781498755238; 9781351216524; 135121652X; 9781351216531; 1351216538; 9781351216548; 1351216546. [Online]. Available: libgen.li/file.php?md5=c58e5e17c764392a3ab5a2c88f571031.

[26] S. Alsenan, I. Al-Turaiki, and A. Hafez, "Feature extraction methods in quantitative structure–activity relationship modeling: A comparative study", *IEEE Access*, vol. PP, pp. 1–1, Apr. 2020. DOI: 10.1109/ACCESS.2020.2990375.

[27] B. Mwangi, T. Tian, and J. Soares, "A review of feature reduction techniques in neuroimaging", *Neuroinformatics*, vol. 12, Sep. 2013. DOI: 10.1007/s12021-013-9204-3.

[28] S. Shalev-Shwartz and S. Ben-David, *Understanding Machine Learning - From Theory to Algorithms*. Cambridge University Press, 2014, pp. I–XVI, 1–397, ISBN: 978-1-10-705713-5.

[29] A. K. Jeremy Watt Reza Borhani, *Machine Learning Refined: Foundations, Algorithms, and Applications*, 1st ed. Cambridge University Press, 2016, ISBN: 9781107123526; 1107123526. [Online]. Available: libgen.li/file.php?md5=a027e3059feeb6b5aa3280951b2bdc20.

[30] S.-M. C. Witold Pedrycz, *Deep Learning: Algorithms And Applications*. Springer International Publishing, 2020.

[31] M. Kothandaraman and A. Pachiyappan, "Comparison of fast ica and gradient algorithms of independent component analysis for separation of speech signals", *International Journal of Engineering and Technology*, vol. 5, pp. 3196–3202, Aug. 2013.

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

[33] M. Scholz and R. Vigário, "Nonlinear pca: A new hierarchical approach", Jan. 2002, pp. 439–444.

[34] L. Weng, "From autoencoder to beta-vae", *lilianweng.github.io/lil-log*, 2018. [Online]. Available: http://lilianweng.github.io/lil-log/2018/08/12/from-autoencoder-to-beta-vae.html.

[35] J. Masci, U. Meier, D. Ciresan, and J. Schmidhuber, "Stacked convolutional auto-encoders for hierarchical feature extraction", Jun. 2011, pp. 52–59, ISBN: 978-3-642-21734-0. DOI: 10.1007/978-3-642-21735-7_7.

[36] A. Payan and G. Montana, "Predicting alzheimer's disease: A neuroimaging study with 3d convolutional neural networks", *ICPRAM 2015 - 4th International Conference on Pattern Recognition Applications and Methods, Proceedings*, vol. 2, Feb. 2015.

[37] A. Makhzani and B. Frey, "K-sparse autoencoders", Dec. 2013.

[38] H. Zou and T. Hastie, "Regularization and variable selection via the elastic net (vol b 67, pg 301, 2005)", *Journal of the Royal Statistical Society Series B*, vol. 67, pp. 768–768, Feb. 2005. DOI: 10.1111/j.1467-9868.2005.00527.x.

[39] A. Hoerl and R. Kennard, "Ridge regression: Biased estimation for nonorthogonal problems", *Technometrics*, vol. 12, pp. 55–67, Apr. 2012. DOI: 10.1080/00401706.1970.10488634.

[40] R. Tibshirani, "Regression shrinkage selection via the lasso", *Journal of the Royal Statistical Society Series B*, vol. 73, pp. 273–282, Jun. 2011. DOI: 10.2307/41262671.

[41] H. Zou and T. Hastie, "Zou h, hastie t. regularization and variable selection via the elastic net. j r statist soc b. 2005;67(2):301-20", *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 67, pp. 301 –320, Apr. 2005. DOI: 10.1111/j.1467-9868.2005.00503.x.

[42] S. Engebretsen and J. Bohlin, "Statistical predictions with glmnet", *Clinical Epigenetics*, vol. 11, Dec. 2019. DOI: 10.1186/s13148-019-0730-1.

[43] E. Gordon, T. Laumann, B. Adeyemo, J. Huckins, W. Kelley, and S. Petersen, "Generation and evaluation of a cortical area parcellation from resting-state correlations", *Cerebral Cortex*, vol. 26, Oct. 2014. DOI: 10.1093/cercor/bhu239.

[44] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A next-generation hyperparameter optimization framework", in *Proceedings of the 25rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2019.

[45] C. Shorten and T. Khoshgoftaar, "A survey on image data augmentation for deep learning", *Journal of Big Data*, vol. 6, Jul. 2019. DOI: 10.1186/s40537-019-0197-0.

[46] H. Zhang, M. Cisse, Y. Dauphin, and D. Lopez-Paz, "Mixup: Beyond empirical risk minimization", Oct. 2017.

[47] L. Isaksson, P. Summers, S. Raimondi, S. Gandini, A. Bhalerao, G. Marvaso, G. Petralia, M. Pepa, and B. Jereczek-Fossa, "Mixup (sample pairing) can improve the performance of deep segmentation networks", *Journal of Artificial Intelligence and Soft Computing Research*, vol. 12, pp. 29–39, Jan. 2022. DOI: 10.2478/jaiscr-2022-0003.

[48] H. Byeon, "Developing a predictive model for depressive disorders using stacking ensemble and naive bayesian nomogram: Using samples representing south korea", *Frontiers in Psychiatry*, vol. 12, p. 773 290, Jan. 2022. DOI: 10.3389/fpsyt.2021.773290.

[49] T. Hastie, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Jan. 2009, ISBN: 9780387848570. DOI: 10.1007/978-0-387-84858-7.

[50] A. Salvalaggio, M. De Filippo De Grazia, M. Zorzi, M. Thiebaut de Schotten, and M. Corbetta, "Post-stroke deficit prediction from lesion and indirect structural and functional disconnection", *Brain*, vol. 143, no. 7, pp. 2173–2188, 2020.

[51] M. Zorzi, M. D. Filippo De Grazia, E. Blini, and A. Testolin, "Assessment of machine learning pipelines for prediction of behavioral deficits from brain disconnectomes", in *International Conference on Brain Informatics*, Springer, 2021, pp. 211–222.