# Alzheimer's Disease Diagnosis using a Variational Autoencoder

Luís Carlos Gonçalves de Castro
luis.carlos.goncalves.de.castro@tecnico.ulisboa.pt

Instituto Superior Técnico, Lisboa, Portugal

November 2019

**Abstract**

Alzheimer's Disease is one of the most common pathologies in aging humans, responsible for progressive brain atrophies which manifest themselves in cognitive and memory problems. Although a form of definitive diagnosis is yet to be found, it is possible to identify disease related lesions through inspection of brain scans, a task that can possibly be automated in order to aid specialists. In this work, a Deep Learning strategy will be adopted in order to build a framework that, given a scan belonging to a patient with an unknown disease status, produces an adequate diagnosis automatically.

Current approaches are all based on the concept of supervised learning through the usage of labeled datasets. This work on the other hand tries to interpret the automated diagnosis of Alzheimer's disease as a novelty detection problem with three classes (healthy, mild cognitive impairment and Alzheimer's dementia), relying on a Variational Autoencoder (VAE), an unsupervised and generative network, modified with convolutional layers responsible for automatic image feature extraction and encoding in the form of a chosen probability density function.

The proposed approach shows advantages, eliminating the need for labelled images and allowing one to inspect affected brain areas (which may be clinically relevant), while showing a possibility to detect anomalies through the output-input reconstruction error. However, it suffers from a highly uninformative latent space and an overdependence on hyperparameter values, which jeopardize its viability, especially when compared to a supervised deep classifier (with the same structure as the unsupervised network's encoder).

**Keywords:** Alzheimer's Disease; Novelty Detection; Variational Autoencoder; Convolutional Neural Network; Variational Inference

## 1. Introduction

This section's goal is to present the problem's motivation, related work and objectives outlined for the proposed approach.

### 1.1. Motivation

Alzheimer's disease (AD) is a progressive brain disorder and one of the most frequent causes of dementia [1], being associated to different levels of cognitive and memory function impairment, which are translated into memory loss, lack of thinking skills or inability to carry out simple tasks. Currently, no cure or mechanism to slow down the progression of the disease has been found, and therefore early diagnosis gathers importance in order to facilitate treatment of symptoms and increase the patient's quality of life.

However, there is no form of definitive diagnosis besides autopsy. In order to indicate the possibility of AD, specialists use neuroimaging, a group of techniques of three-dimensional brain *in vivo* images that have the purpose of studying the structural anatomy or activity of the brain. This facilitates the existence of computerized approaches that, using these images, are able to automatically detect disease patterns even when symptoms are not yet fully visible[2], serving as a tool that can aid a specialist's diagnosis endeavour.

Handling neuroimages on the other hand is not trivial due to the complexity and variability of the human brain, the size of the images itself and the amount of samples which are needed in order to build an automatized tool. So, this work follows a Deep Learning (DL) strategy, a branch of algorithms popular for their ability to handle large and complex datasets and essentially learn key features without extensive preprocessing[3][4]. They also open the possibility for one to limit the amount of labelled data used to train such a tool, which in practice would spare resources (images would not need to be classified *a priori* by specialists).

### 1.2. Related Work

To understand the extensive list of work already done in this field, it's crucial to notice that different approaches may not only focus on detecting certain phases of the disease, according to the subject's medical status, but also use different neuroimaging types, feature extraction methods (either automatic or using *a priori* chosen regions of interest (ROI)) and networks.

Some works try to distinguish two phases: Normal Control (NC), characterized by subjects who are either healthy or show natural age-related decline, and Alzheimer's Disease (AD), comprised of patients whose atrophy is severe and irreversible. Examples of this are a Convolutional Neural Network (CNN) proposed by Bäckström et al.[5] that automatically extract features from Magnetic Resonance Images (MRI) in order to classify subjects or Zheng et al.[6] proposal of a Deep Polynomial Network (DPN) that fuses ROI intermodal information from both MRI and Positron-Emission Tomography (PET) images, then using a Support Vector Machine (SVM) to classify images.

Additionally, other DL approaches may also take into account the intermediate phase of Mild Cognitive Impairment (MCI), in which patients show signs of brain changes which are bigger than those of NC subjects, but not as severe as AD. This is the case of the CNN's proposed by Vu et al.[7] and Aderghal et al.[8] which automatically extract features, with the first using a sparse autoencoder (SAE) to isolate relevant information in a first stage and the latter focusing on feature detection using aligned ROI's of two different image modalities.

Since MCI may or may not progress into AD in a specific time interval, some works (e.g. the ROI based Deep Boltzmann Machine (DBM) solution of Suk et al.[9] and the automatic feature extraction CNN proposed by Choi et al.[10]) divide this group into two different classes: non-converting MCI (MCI-NC) and converting MCI (MCI-C), which itself can be seen as a forecasting problem, with the proposed networks capturing subtle information that a patient may in the future develop AD.

### 1.3. Main Goals

Almost all proposed works so far in the field of automated AD diagnosis using DL are based on the concept of supervised learning, which requires that every training sample is labelled. This thesis' aim is to build a classifier that works off a fully unsupervised and generative solution which requires no labels during training (using a novelty detection approach), and then is able to identify three different classes: NC, MCI and AD. Further, it is intended to extend the MCI class into MCI-C and MCI/NC, considering different conversion time windows in which the patient may or may not progress into an AD state. In order to compare the performance of the given solution, a supervised DL classifier with a similar structure to the unsupervised network will be assembled, tested and used as a baseline to compare with the unsupervised-based solution.

## 2. Background

This section aims at introducing theoretical concepts such as fully connected and convolutional neural networks, autoencoders and variational autoencoders.

### 2.1. Fully Connected Networks

Fully connected neural networks (FC networks) [3][4] are DL models which are constituted by consecutive layers whose nodes are all connected with each other with specific weights $w$ and bias $b$. Defining two consecutive layers as $l^{(k-1)}$ and $l^{(k)}$ with $n$ and $m$ units respectively, one can describe an output-input model based on a weight matrix $w^{(k)} \in \mathbf{R}^{m\times n}$ and a bias vector $b^{(k)} \in \mathbf{R}^{n\times 1}$ as

$$o^{(k)} = \begin{cases} x \text{ if } k = 0 \\ \theta\big((o^{(k-1)})^T w^{(k)} + b^{(k)} \text{ if } k \geq 1\big) \end{cases}, \quad (1)$$

in which $\theta(.)$ is a non-linear activation function. FCNN's, as other networs belonging to the DL branch, may have several layers which allow for a sequential and hierarchical extraction of higher level features from an initial input. FCNN's are optimized during training in order to minimize a loss function $\mathcal{L}(\theta)$ that can assume different forms depending on the problem at hand, like for instance the binary or categorical cross-entropies in classification problems that try to minimize the discrepancy between binary coded labels and the output class probabilities. The update of $w$ and $b$ is done using backpropagation methods such as stochastic gradient descent[11], which are based on the computation and minimization of the loss function's gradient which is dependant of those same parameters.

### 2.2. Convolutional Neural Networks

Convolutional Neural Networks (CNN)[3][4] are models based on the concept of receptive fields, mainly focused on capturing local information hierarchically: starting from low level and local features, which are later combined and extracted into higher level ones. This is made possible through the use of convolutions, formally defined by

$$s[n] = (f * g)[n], \quad (2)$$

in which $f$ is the multidimensional input, $g$ is a convolution filter (also called kernel) which captures different local features across different positions of $f$ and $s$ is the resulting feature map, expressed in a lower dimensional space as compared to the original input space. By assembling and setting different kernels with different weights and applying an activation function to $s$, one can form a convolutional layer, in which those values are regarded as network parameters that can be learned during training, being its multi-dimensional size a hyperparameter. CNN's are therefore a set of sequential convolutional layers, presenting usually a FC network in the end responsible for combining the learnt high level features and producing an output. These layers may also present other non-linear operations such as pooling or strided-convolutions, which contribute to keeping the extracted feature's size manageable by downsampling the obtained feature maps[3].

CNN's share with FC networks the same method of training and objectives, with the difference of having a much sparser approach by assuring that $g$ is smaller than $f$ and by sharing kernel weights, which minimizes the amount of trainable parameters. Besides this, convolutions are translation invariant due to the kernel weights being independent of the input's position[3].

### 2.3. Transposed Convolutional Neural Networks

In the case of CNN's, convolving an input and a kernel can be expressed in terms of a matrix multiplication between a flattened version of the first and a convolution matrix $m$, in which every row defines one single convolution operation. Oppositely, certain applications such as image denoising or interpolation require feature maps to be defined in bigger dimensions, thus giving room to the use of transposed convolution networks (TCNN) [12]. Similarly, transposed convolutions may be seen as a multiplication between a transposed version of the convolution matrix, $m^T$, and a flattened input, thus resulting in a larger one-dimensional feature map $p$ being assembled.

Conceptually, as a standard convolution maps a many-to-one relationship that is imposed between input and output feature map, a TCNN may be seen as a stack of upsampling layers that enforce one-to-many relationships, mapping single values into receptive fields.

### 2.4. Autoencoders

An autoencoder (AE)[3][4] is a neural network that is trained in order to copy an input to an output, by compressing the first into a lower dimension space (through the use of an encoder neural network) and then converting the latter back to its original space (using a decoder neural network). Formally, an autoencoder is defined by

$$\hat{x} = g(z) = g(f(x)), \tag{3}$$

in which $x$ is the input, $f(.)$ is the encoding function that compresses the input into a latent variable $z$ (represented in a lower dimensional space), and $g(.)$ is the decoding function that taking $z$ as an argument produces a reconstruction of the input $\hat{x}$. By lowering the dimensions of the encoded space representation, the AE will only be able to output an approximate replica of the input; however, this compression means that the network learns how to prioritize certain features over less relevant ones, thus learning powerful and underlying properties about the data in hand.

The AE is an unsupervised neural network, being trained in order to minimize a cost function $\mathcal{L}(x, \hat{x})$ that evaluates the degree of dissimilarity between input and output (L2-loss function, for instance).

### 2.5. Variational Autoencoders

A variational autoencoder (VAE)[13] is a generative and unsupervised neural network which is based on the concepts of variational inference and variational expectation maximization. The VAE works with the premise that any training set $x$ can be generated from any underlying unobserved representation $z$ according to an *a priori* chosen density function $p(z)$ (e.g. $\mathcal{N}(0, I)$, which is very common in practice[13]). This is equivalent to the computation of latent variables $z$ that are more likely to generate samples similar to the examples in $x$ by means of a function $\phi$ (i.e. computing $p_\phi(z|x)$), which is in practice intractable[13].

Just like AE's, VAE's are composed of two networks: an encoder and a decoder (parametrized by $\theta$ and $\phi$, respectively), with the first focusing on solving the intractability problem by approximating $q_\theta(z|x)$ to $p_\phi(z|x)$ and producing probability density function (PDF) parameters (in the case of a Gaussian Distribution, $\mu, \Sigma$) that will be similar to those of the *a priori* chosen $p(z)$, a method known as variational inference. On the other hand, the decoder aims at fulfilling the original task: from an latent input, generate a sample $\hat{x}$ which likely is similar to the samples in $x$, a step known as variational expectation maximization. This objective may be expressed as the maximization of the Evidence Lower Bound (ELBO)

$$\begin{aligned} ELBO = E_{q_\theta(z|x)}&[log p_\phi(x|z)] \\ &- D_{KL}(q_\theta(z|x)||p_\phi(z)), \end{aligned} \tag{4}$$

in which the first term encourages the output to be similar to the input in order to maximize the expected value of the likelihood of the data samples $x$ given the latent code $z$, and the second is the Kullback-Leibler (KL) divergence that forces the latent distribution $q$ obtained by the encoder to approximate a target distribution $p(z)$.

In order to train the VAE (and assuming that the *a priori* p(z) is a multivariate Gaussian PDF), one has to take into account the stochastic node in the network, responsible for sampling the latent variable $z \sim N(\mu, \Sigma)$, which denies the possibility of using any form of backpropagation. However, in this specific case, it's possible to decompose $z$ as

$$z = \mu + \Sigma \times \epsilon, \ \ \epsilon \sim N(0, I). \tag{5}$$

in which $\epsilon$ is an auxiliary stochastic variable. The decomposition in eq.5 is known as reparametrization trick, and it opens a deterministic path in the network for the gradient to be calculated in successive fashion, reducing the stochasticity to a multivariate unit Gaussian noise injection in the network.

## 3. Methodology

In this section, the proposed approach is presented: this includes the supervised deep classifier and the proposed unsupervised variational autoencoder based pipeline for AD classification.

### 3.1. Convolutional Neural Network Classifier

The supervised network will work as a feature extractor and a classifier, labelling input neuroimages. The structure of the CNN considers two parts: a feature extraction part, composed of convolutional and pooling layers, which facilitate high-level learning of image patterns with a reasonable amount of parameters, and a classification part, composed of fully connected layers, as presented in figure 1. The assumed distribution $p(z)$ is a multivariate Gaussian with unit variance and zero mean.

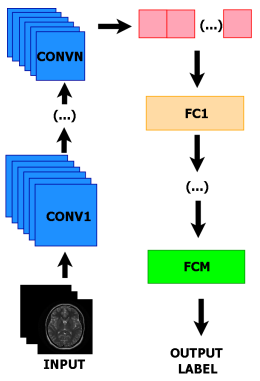


**Figure 1:** Architecture of the CNN used for classification purposes.

This network will be applied to several binary problems of classification, such as:

- AD/NC, distinguishing healthy subjects from Alzheimer's Dementia patients;
- AD/MCI-NC, setting apart two anomalous states that indicate a bigger than expected mental health decline (patients that eventually convert to AD are not considered);
- MCI-NC/MCI-C, which can be seen as a forecasting test, where the network tries to model specific patterns that imply that a patient with MCI will eventually progress into AD or not. This problem is further divided considering different conversion time windows (immediate and 12 or 24 months).

Taking into consideration the binary nature of the labelling problems in hand, the loss function is:

$$\mathcal{L}(\theta) = \sum_{i=1}^{N} \Big( y_i \log(p(\theta)) + (1 - y_i) \log(1 - p(\theta)) \Big), \tag{6}$$

in which $\theta$ is the matrix of weights of the network, $p(\theta)$ is the produced probability parametrized by the network parameters of the input data $x$ belonging to a certain class $y$.

### 3.2. Convolutional Variational Autoencoder

In this subsection, the framework used for AD classification whose core is an unsupervised variational autoencoder is presented.

#### 3.2.1 Proposed Network

The core of the unsupervised approach lies in the use of a convolutional VAE, with the encoding and decoding networks assuming the form of a CNN and a TCNN, respectively. In order to compare both supervised and unsupervised networks, the encoder will be as similar to the CNN as possible. The general architecture of the VAE is presented in figure 2. Assuming that the samples within the dataset $x$

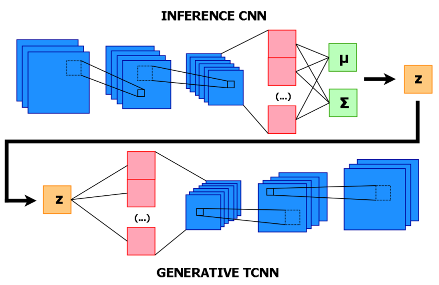


**Figure 2:** Proposed architecture of a convolutional variational autoencoder.

are indepedent and identically distributed, the loss function one intends to minimize assumes the form of the sum of the negative ELBO's computed for each sample $x_i$, formally as

$$L_{VAE} = \sum_{i=1}^{N} \Big[ - E_{q_\phi(z|x_i)}[logp_\theta(x_i|z)] + \beta D_{KL}(q_\phi(z|x_i)||p(z)) \Big], \tag{7}$$

in which $\beta$ is a parameter that balances the input reconstruction capability and the latent-space distribution constraint[14]. This constant follows an annealing scheme in order to achieve proper latent disentanglement and avoid uninformative latent codes[15][16]: starting as 0 in the first training epoch and growing until reaching it's final value in a certain epoch, remaining fixed until the end of the network's tuning. The negative expected log likelihood of the data (first term in eq. 7) is implemented as the output's mean squared error when trying to generate a sample likely similar to the input.

#### 3.2.2 Novelty Detection Modes

The proposed approach is based on the notion of novelty detection[17][18], meaning that the VAE will only be trained using NC samples, learning to encode specific healthy brain features into the desired PDF and consequently use them to generate an output likely similar to the neuroimages belonging to the single class training dataset. Post-training, the outlier patterns (corresponding to anomalous images, either MCI or AD) may be detected in several forms:

- by analyzing the latent feature space (e.g. Bhattacharyya distance), in which it is expected that both MCI and AD samples produce distribution parameters that diverge more from the *a priori* chosen $p(z)$ than the PDF's produced by a NC image;
- evaluating the original input space (e.g. mean squared error), where the reconstruction error committed by unseen NC samples is expected to be lower than the one produced by any anomalous class;
- using the VAE loss as a metric, theoretically smaller for any NC image due to the fact that is the only class represented in the training dataset.

MCI samples are expected to produce scores which are closer to NC samples when compared to AD images. The same rule applies for MCI-C and MCI-NC samples, with the latter corresponding to a state in which the brain is closer to being healthy than to develop dementia as compared to the first. The decision thresholds are learned through a classifier or by averaging the difference of mean scores obtained for every neighbour classes using a small image subset

## 4. Results & discussion

In this section the obtained results for both supervised and unsupervised architectures are presented.

### 4.1. Dataset and Experimental Setup

The dataset used in this work has been obtained from the Alzheimer's Disease Neuroimaging Initiative (ADNI), and consists of Fluorodeoxyglucose Positron Emission Topography (FDG-PET) functional neuroimages from subjects belonging to NC, MCI and AD classes, each one having a maximum of three brain scans that correspond to different timestamps of a continuous study (Baseline month, 12$^{th}$ month and 24$^{th}$ month) in which their disease status maychange. In table 1, a summary of the existing images is given, along with gender information, age and Mini-Mental State Exam (MMSE) average scores and respective standard-deviation (in both, it is assumed these values assume the form of a Gaussian distribution).

| | **NC** | **MCI** | **AD** |
|---|---|---|---|
| **Images** | 271 | 526 | 228 |
| **Gender** | 180/91 | 346/180 | 136/92 |
| **Age - $\mu(\epsilon)$** | 76.96(4.82) | 76.02(7.34) | 76.64(7.05) |
| **MMSE - $\mu(\epsilon)$** | 29.11(1.14) | 26.57(2.87) | 21.43(4.58) |
| **Subjects** | | 406 | |

**Table 1:** FDG-PET dataset summary table.

The inter-class age difference is small, assuring that the solution captures differences in between classes due to the disease status and not age differences; on the other hand, the low intra-class age standard deviation avoids heterogeneous samples, guaranteeing that the proposed networks capture class relevant features and not age specific ones. One of the goals of this thesis is also to solve MCI-C/MCI-NC problem assuming several time windows. For this problem, the number of images are presented in table 2.

| | **MCI-NC** | **MCI-C** |
|---|---|---|
| **Images (Immediate)** | 456 | 70 |
| **Images (12 months)** | 400 | 126 |
| **Images (24 months)** | 366 | 166 |

**Table 2:** Number of images for the MCI-NC/MCI-C problem for different time windows.

The neuroimages (of original size 128x128x60) are cropped (skull-stripping), normalized ( intensity values in between a range of 0 and 1) and afterwards padded with zeros, being its final size of 64x80x64 pixels. Since every patient may have more than one scan (corresponding to different timestamps), the data partitioning follows a subject-separating strategy: this means that it is assured that scans of a different subject are guaranteed to not occupy the ideally independent training and test sets.

All proposed architectures and models have been implemented using the Python-based Machine Learning library Keras (version 2.2.4) with Google's TensorFlow backend (version 1.13.1). Experiments have been conducted on a workstation with an Intel-i7 8700 3.20GHz CPU, 16GB RAM and a NVIDIA-GTX 1080 11GB GPU.

### 4.2. Supervised Classifier Network

It is intended to first solve the problem using a supervised framework, serving as a baseline performance metric for the case in which no labels are used during the network's training, following the method described in section 3.2. In this section, the network and different results are presented.

#### 4.2.1 Training Settings and Hyperparameters

The network's chosen architecture is characterized in table 3. The data used is partitioned following a subject-separation criteria (80% of dataset is used for training, 10% for validation and 10% for testing), and its performance is evaluated using 10 fold cross-validation. The network is trained to minimize the binary cross-entropy loss function and uses the Adam optimizer, with initial learning rate $\eta_0 = 0.0001$,$\beta_1 = 0.9$ and $\beta_2 = 0.999$. The network training lasts for 50 epochs (80 in MCI-C/MCI-NC problems) and has a mini-batch size of 40 images per iteration. An early stopping strategy is adopted, in which the final network parameters correspond to the training epoch that has resulted in the best performance in the validation set.

| Layer | Specifications | Units |
|---|---|---|
| **Convolutional** | Same Padding, Stride-2, ReLU, MaxPool 2x2x2 | 5x5x5x32 |
| **Convolutional** | Same Padding, ReLU MaxPool 2x2x2 | 5x5x5x64 |
| **Convolutional** | | 5x5x5x128 |
| **Convolutional** | | 3x3x3x256 |
| **Fully Connect** | ReLU, 50% dropout | 512 |
| **Fully Connect** | ReLU | 128 |
| **Output** | Sigmoid | 2 |

**Table 3:** Architecture of the supervised classification network. (Conv. - Convolutional Layer; FC - Fully Connected Layer)

In table 2 its noticeable that the MCI-C/MCI-NC problems is class imbalanced[19], an therefore a weighted training strategy is applied, in which samples belonging to the dominant class with $N$ samples are given unitary weight and samples belonging to the smaller class (with $M$ samples) are entitled to an $\frac{N}{M}$ weight [19].

#### 4.2.2 Results using the Original Dataset

The results obtained with the proposed CNN are presented in table 4, being expressed in terms of accuracy (ACC), specificity (SPEC), sensitivity (SENS) and the F1-Score (F1). Some state of the art results are presented in table 5, in order to compare the supervised network's performance. Considering the NC/AD problem first, the proposed network reaches a mean accuracy of 89,25% with 91,06% rate of well identified Alzheimer's Disease cases, a value that is comparable to cutting edge results but overall slightly lower when compared for instance with the results of Bäckström et al. in [5] which achieve 90,1% accuracy and 93,26% sensitivity or the ROI aided solution of Zheng et al. in [6] with 97,27% accuracy.

| Classes | ACC | SPEC | SENS | F1 |
|---|---|---|---|---|
| **NC/AD** | 0.8925 | 0.8700 | 0.9106 | 0.8877 |
| **MCI/AD** | 0.8456 | 0.8780 | 0.7510 | 0.8094 |
| **MCI-C/MCI-NC(I)** | 0.8400 | 0.8400 | 0.8400 | 0.8400 |
| **MCI-C/MCI-NC(12)** | 0.7325 | 0.7250 | 0.7372 | 0.7310 |
| **MCI-C/MCI-NC(24)** | 0.6750 | 0.6700 | 0.6759 | 0.6729 |

**Table 4:** Classifier results for different classification problems. (I - Immediate Conversion; 12 - 12 months conversion window; 24 - 24 months conversion window)

The results in general also confirm the difficulty that is solving the problem of mild cognitive impairment conversion (MCI-NC/C problem) when compared to, for instance, the simpler problem NC/AD, with the CNN being unable to reach accuracy or sensitivity rates above the 84% mark (obtained for an immediate conversion). Increasing the conversion time window for 12 months, an accuracy of 73,25% and F1-score of 73,1% are obtained, slightly lower than those of Suk et al. [9]. For the longest term prevision of 2 years (24 months), an accuracy of 67,5% and sensitivity of 67,59% is produced, lower than the results of Choi et al. in [10] that present for a larger conversion period of 36 months an accuracy of 84,2% and specificity of 87,0%.

| Authors | Problem | ACC | Dataset |
|---|---|---|---|
| **Bäckström [5]** | NC/AD | 90,10% | 199(NC)/141(AD) |
| **Zheng et al.[6]** | NC/AD | 97,27% | 52(NC)/51(AD) |
| **Vu et al.[7]** | MCI/AD | 85,00% | 399(MCI)/188(AD) |
| **Suk et al. [9]** | MCI(C/NC) (24 months) | 75,92% | 204(MCI) |
| **Choi et al. [10]** | MCI(C/NC) (36 months) | 84,2% | 79(MCI-NC)/92(MCI-C) |

**Table 5:** Main results obtained for state of the art approaches

#### 4.2.3 Results using the Augmented Dataset

In order to improve the quality of the network, one can perform data augmentation to the dataset. Since AD is a disease that affects the brain as a whole and due to the symmetry the brain presents, flipping left and right hemispheres may work as a variation of the original neuroimages and thus improve performance, increasing the number of training examples (in this case, doubling the number of available images). In comparison with table 4, one

| Classes | ACC | SPEC | SENS | F1 |
|---|---|---|---|---|
| **NC/AD** | 0.9100 | 0.9050 | 0.9132 | 0.9090 |
| **MCI/AD** | 0.8600 | 0.8875 | 0.8352 | 0.8601 |
| **MCI-C/MCI-NC(I)** | 0.8475 | 0.8400 | 0.8459 | 0.8479 |
| **MCI-C/MCI-NC(12)** | 0.7875 | 0.7850 | 0.7891 | 0.7870 |
| **MCI-C/MCI-NC(24)** | 0.7675 | 0.7650 | 0.7678 | 0.7664 |

**Table 6:** Classifier results for different classification problems with an augmented dataset (I - Immediate Conversion; 12 - 12 months conversion window; 24 - 24 months conversion window) .

can immediately spot that performance increases. For instance, the NC/AD labelling problem now achieves a 91% accuracy, 91,32% sensitivity, 90,5% sensitivity and 90,9% F1-score, results that slightly tops the one achieved by Bäckström et al. in [5] but still inferior to other solutions like the ROI based solution of Zheng et al. in [6]. The conversion problems also increase their performance metrics, with the 24-month case seing the biggest increase, reaching an 76,75% accuracy and 76,64% F1-score, barely topping the 18 month conversion classifications of Suk et al. [9] but still slightly below the approach proposed by Choi et al. [10]. In this problem specifically, data augmentation also tackles directly the class imbalance problem by increasing the number of samples that represent the less populated class (MCI-C), and thus can be seen as one of the possible reasons for improving the obtained results.

## 4.3. Variational Autoencoder

In this part, the novelty detection classifier using an unsupervised VAE is presented and the results are showcased. There were problems in the latent space disentanglement and the *trade-off* between this and the reconstruction ability, which will be further explained, along with experiences and scenarios that tried to (unsuccessfully) solve the issue.

### 4.3.1 Training Settings and Hyperparameters

Again, the data split follows a subject separation criteria, in which only normal control (NC) images are considered during training. From the datapoints available in this class, 80% are saved for the training set, 10% for the validation set and 10% for testing. The train set is also augmented by rotating the brain (flipping left and right hemispheres). Again, the Adam optimizer is used to train the network (with initial learning rate $\eta_0 = 0.0001$, $\beta_1 = 0.9$ and $\beta_2 = 0.999$). Training lasts for 450 epochs and the mini batch is of size of 12. Again, early stopping is applied with the best performance in the validation set (smallest loss value) as criteria. The latent space has size $k = 128$ and the parameter $\beta$ responsible for the balance between the reconstruction term and KL-divergence follows an annealing schedule, reaching the final value of 0.01 in the 100th epoch which is kept constant for the rest of the training iterations.

| Inference Network (Encoder) | | |
|---|---|---|
| **Layer Type** | **Specifications** | **Filters/Units** |
| **Convolutional** | Same Padding, Stride-2, ReLU, MaxPool 2x2x2 | 5x5x5x32 |
| **Convolutional** | Same Padding, ReLU MaxPool 2x2x2 | 5x5x5x64 |
| **Convolutional** | | 5x5x5x128 |
| **Convolutional** | | 3x3x3x256 |
| **Fully Connect** | ReLU, 50% dropout | 512 |
| **Fully Connect** | ReLU | 128 |
| **Output** $(\mu, \epsilon)$ | Linear | 64 |

| Generative Network (Decoder) | | |
|---|---|---|
| **Layer Type** | **Specifications** | **Filters/Units** |
| **Sampling** | Gaussian PDF Sampling Layer | — |
| **Fully Connect** | ReLU | 128 |
| **Fully Connect** | ReLU | 512 |
| **Deconvolutional** | Same Padding, ReLU MaxPool 2x2x2 | 3x3x3x256 |
| **Deconvolutional** | | 5x5x5x128 |
| **Deconvolutional** | | 5x5x5x64 |
| **Deconvolutional** | ReLU (deconv) | 5x5x5x32 |
| **+ Compression** | Sigmoid (compression) | 5x5x5x1 |

**Table 7:** Architecture of the unsupervised Variational Autoencoder

### 4.3.2 Results

The training performance for the test set is presented in figure 3, showing that the VAE converges to an assumed optimal solution that minimizes the cost function.

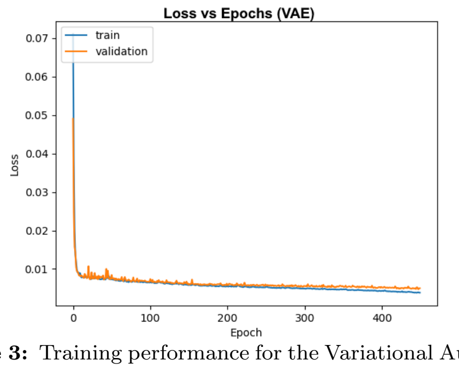


**Figure 3:** Training performance for the Variational Autoencoder.

The next step is now to input images from the test set, which include samples belonging to anomalous classes (MCI and AD) and analyze not only the reconstruction performance but also the latent distribution parameters originated by the encoder. The reconstruction errors are calculated using a pixel mean squared error (MSE) between input and network output, and the results are presented in figure 4. By training the VAE only with NC samples,

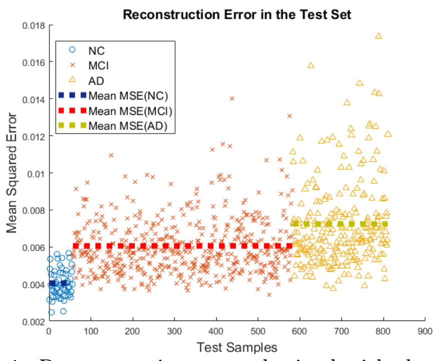
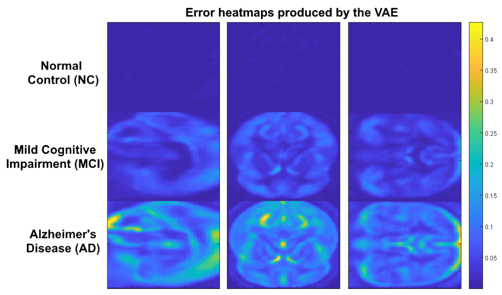


**Figure 4:** Reconstruction error obtained with the test set: **(b)** individual sample MSE with the average class MSE signed with a line. Blue data represents the NC samples, red MCI and yellow AD.

the network learns to generate images that are likely similar to those in the training set. Thus, it would be expected that both anomalous classes (MCI and AD) would present higher reconstruction errors and that MCI images would present a MSE smaller than its counterpart AD. In figure 4(b), it can be seen that the class average MSE's follow what would be expected, with the average MSE for the AD class being the biggest and the NC the smallest, something that is confirmed by the mean absolute error (MAE) heatmaps shown in figure 5. In the different



Error heatmaps produced by the VAE
Normal Control (NC)
Mild Cognitive Impairment (MCI)
Alzheimer's Disease (AD)

**Figure 5:** Error heatmaps produced for every class by the VAE in three different brain slices, with the decimal scale that indicates the range of committed mean absolute error.

ent 2D slices of the heatmaps produced for each class, areas with a bigger reconstruction error are more noticeable in the anomalous classes (MCI and AD), thus implying that the network will not be able to proficiently generate anomalous features. In figure 6, regions of interest (ROI) which are medically relevant for the diagnosis of AD are shown in the same three 2D planes where the heatmaps are represented.

By comparing the ROIs and heatmaps, it's visible that some areas with bigger error that were captured by the reconstruction heatmaps are indeed medically relevant. With this, both results in figures 4 and 5 show two clear advantages in practical terms: not using any labels during training and and specifying information about which brain areas the VAE failed to reconstruct (something that may be associated to disease related pathologies).

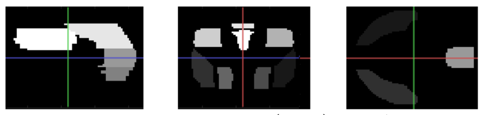


**Figure 6:** Regions of Interest (ROI) for Alzheimer's Dementia Diagnosis. The three slices correspond to the same three planes where the heatmaps are depicted in figure 5.

However, these conclusions are all made out of average measurements. Again, analyzing the sample-reconstruction errors in figure 4(a), one can notice that a considerable quantity of NC images produce reconstruction error values that are very close to the mean MSE produced by samples of the MCI class. The same conclusion can be applied to some MCI class samples that also present values equal or above the average AD sample mean squared error, something that jeopardizes the efficiency of a classifier whose task is defining decision thresholds for the errors committed in every class and eventually labelling an unknown neuroimage.

In order to improve the results, it's important to remember that the decoder uses a multidimensional variable generated by the parametrized PDF (encoder output) as an input. Here, its expected that anomalous samples will fail at matching the *a priori* $p(z)$, producing parameters which are distant from the values normally produced in a training phase (equivalent to healthy samples). To evaluate this (with the same settings described in 7), and with the aid of dimensional reduction algorithm t-Distributed Stochastic Neighbor Embedding (t-SNE) one can visualize the produced latent-space PDF parameters for the test set, as shown in figure where the means are plotted in a 2D space.

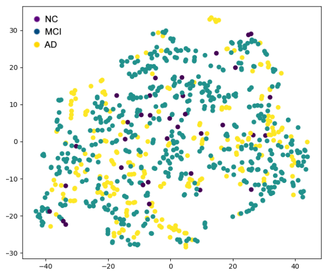


**Figure 7:** Visualization of the high-dimensional latent generated means using t-SNE. Purple, navy blue and yellow dots are generated by NC, MCI and AD image, respectively.

The figure 7 shows an unstructured latent space, revealing that the network is unable to form clusters or to, when exposed to a different class of images compared to the training set (in this case, MCI or AD), produce mean values that are substantially different from those which are generated by NC images. This suggests that the reconstruction errors shown in figures 4 and 5 are affected by the natural brain inter-class variance.

One may also tune the parameter $\beta$, which embodies the trade-off between input reconstruction and latent space structure. Since the latter results show that the latent space suffers from lack of disentanglement, an experiment in which $\beta$ is increased from 0.01 to 0.1 is shown in figure 8. Increasing

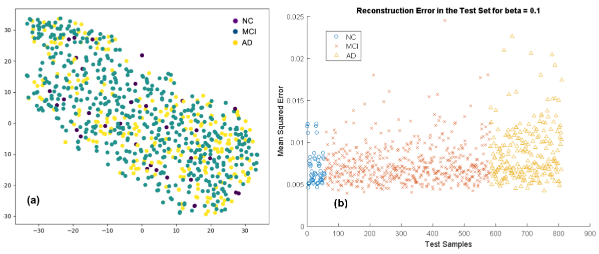


**Figure 8:** Results for VAE with $\beta = 0.1$: visualization of the high-dimensional latent generated means using t-SNE(a) and the reconstruction MSE(b)

the value of $\beta$ actually had the opposite effect to what would be expected: approximating the latent means without significant latent disentanglement. Additionally, the reconstruction MSE's are now very similar in between classes, what is explained due to the latent PDF parameters being input independent and thus generating very similar outputs. An example is given in figures 9, in which same 2D cuts of three different class neuroimages originate very similar and almost indistinguishable outputs, what can be perceived as underfitting.

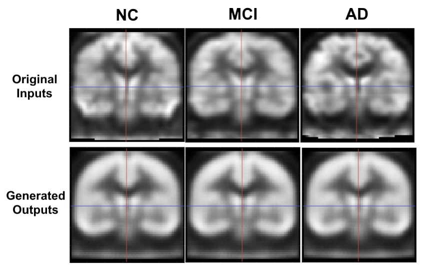


**Figure 9:** Example of three different 2D slices of generated neuroimages with different inputs, where underfitting is apparent.

Another option would be to decrease the dimensions $k$ of the latent space, a hyperparameter whose value deeply affects a variational autoencoder and mainly its ability to reconstruct the input: if $k$ is too big, reconstructions may become class independent due to the lack of feature selection made by the encoder; if $k$ is too small, the data compression enforced in the encoder is too harsh and can lead to sub-optimal reconstructions. For $\beta = 0.01$, several values of $k$ were experimented, with results shown in figure 10. From the plots presented in figure 10, no major differences in terms of disentanglement are achieved. There are several combinations of $\beta$ and $k$ values which were for the most tested but

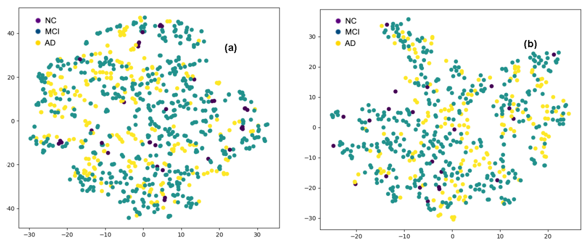


**Figure 10:** Latent means generated by the encoder for latent space dimensions of $k = 32$ (a) and $k = 256$ (b)

all presented results are similar to what was been shown earlier and do not add anything relevant.

Also, from a practical standpoint it's important to compare the memory and time consumption between the proposed supervised and unsupervised networks, as shown in table 8, where it's noticeable that the VAE drains more resources than the CNN, by doubling the number of layers, having more than the triple of the parameters and almost quadrupling the required memory for every input image, which can harm the performance of the network and require a much larger amount of training samples.

| | **CNN** | **VAE** |
|---|---|---|
| **Number of Layers** | 12 | 24 |
| **Number of Parameters** | 6 954 242 | 22 578 689 |
| **Memory Required per Image during Training (megabytes)** | 44 MB | 183 MB |
| **Average Training Time** | 1 hour | 9 hours |

**Table 8:** Comparison of the two networks in terms of size, memory required (per image) and the average training time using the horizontal mirroring augmented dataset.

## 5. Conclusions and Future Work

Comparing the CNN and the convolutional VAE-based approaches, one can argue that the first is superior in terms of performance. The VAE struggles a lot with hyperparameter tuning, being sensitive to changes with the latent space dimensionality $k$ and the regularizing parameter $\beta$ which control the structuring of the latent space as opposed to the rate of error committed in reconstructions, whose balance is key to achieve success using a similar method to this one. The problems with unstructured latent spaces and sub-optimal reconstructions, according to recent studies[15][16], may be associated to uninformative latent codes (which the decoder may ignore) and vanishing KL-terms. Additionally, this trade-off is still poorly misunderstood in practice, with the hyperparameter $\beta$ being a lot of times defined in an empirical way rather than analytically. Also, an additional difficulty is the huge computational load and very high training and testing time per simulation imposed by this network.

However, this approach brings a smaller dependency on labelled datasets as the majority of state of the art solutions present, due to the unsupervised nature of the VAE. This approach is also user-friendly, and ideally may indicate in which brain regions the disease is noticeable, something that may aid diagnosis in an intuitive way. Additionally, the VAE is a generative network: with proper tuning, it can serve also as an augmentation mechanism to increase the dataset's size and exponentialize performance pre-existing solutions.

For any future work related with this thesis, one has to assess the main problems explained before that this approach faces (especially when exposed to neuroimages). Some future lines of work may pass by: applying adaptable methods that change the value of $\beta$ to this problem may function to increase meaningfulness in the latent code instead of empirically adjusting it, changing the chosen distribution from the usually chosen multivariate unit Gaussian (due to its simple and uninformative nature) and altering or varying the objective loss function of the network. Also, a semi-supervised version of the problem which basically consists of adding a classifier network to the encoder and use the produced labels as a part of the latent space to increase latent code relevancy may be taken into account[20].

**References**

[1] Alzheimer's Association (2018): "Alzheimer's Disease Facts and Figures". *Alzheimer's & Dementia: The Journal of the Alzheimer's Association*, *14*(3).

[2] B.C. Dickerson et al. (2011): "Alzheimer-signature MRI biomarker predicts AD dementia in cognitively normal adults". *Neurology*, 76(16).

[3] I. Goodfellow et al. (2016): "Deep learning" (Vol. 1). *Cambridge: MIT press*

[4] F. Chollet (2017): "Deep learning with python". Manning Publications Co.

[5] K. Bäckström et al.(2018): "An efficient 3D deep convolutional network for Alzheimer's disease diagnosis using MR images". *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI)*, pp. 149–153.

[6] X. Zheng et al.(2016): "Multi-modality Stacked Deep Polynomial Network based Feature Learning for Alzheimer's Disease Diagnosis". 2016 IEEE 13th International Symposium on Biomedical Imaging (ISBI), pp. 851-854.

[7] T.D. Vu et al.(2017): "Multimodal learning using convolution neural network and Sparse Autoencoder". *2017 IEEE International Conference on Big Data and Smart Computing (BigComp)*, pp. 309-312. IEEE.

[8] K. Aderghal et al.(2018): "Classification of Alzheimer Disease on Imaging Modalities with Deep CNNs Using Cross-Modal Transfer Learning". *2018 IEEE 31st International Symposium on Computer-Based Medical Systems (CBMS)*, pp. 345-350. IEEE.

[9] H.I. Suk et al.(2014): "Hierarchical feature representation and multimodal fusion with deep learning for AD/MCI diagnosis". *NeuroImage, 101*, pp. 569-582.

[10] H. Choi et al.(2018): "Predicting cognitive decline with deep learning of brain metabolism and amyloid imaging." *Behavioural brain research*, 344, 103-109.

[11] D.E. Rumelhart et al.(1988): "Learning representations by back-propagating errors." *Cognitive modeling*, 5(3), 1.

[12] M. D. Zeiler et al.(2010): "Deconvolutional networks." *IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2010)*, pp. 2528-2535. IEEE.

[13] D.P. Kingma et al.(2013): "Auto-Encoding Variational Bayes." *arXiv preprint* arXiv:1312.6114.

[14] I. Higgins et al.(2017): "beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework". *5th International Conference on Learning Representations (ICLR)*, *2*(5).

[15] S.R. Bowman et al.(2015): "Generating sentences from a continuous space". *arXiv preprint* arXiv:1511.06349.

[16] X. Liu et al.(2019): "Cyclical Annealing Schedule: A Simple Approach to Mitigating KL Vanishing." *arXiv preprint* arXiv:1903.10145.

[17] A. Vasilev et al.(2018): "q-Space Novelty Detection with Variational Autoencoders". arXiv preprint, arXiv:1806.02997.

[18] Y. Lu et al.(2018): "Anomaly detection for skin disease images using variational autoencoder". *arXiv preprint*, arXiv:1807.01349.

[19] S. Wang et al.(2016): "Training deep neural networks on imbalanced data sets." *International Joint Conference On Neural Networks (IJCNN 2016)*, pp. 4368-4374

[20] D.P. Kingma et al. (2014): "Semi-supervised Learning with Deep Generative Models." *Advances in neural information processing systems*, pp. 3581-3589.