TÉCNICO LISBOA

Modelling Progression of Alzheimer's Disease with RNN

Pedro Miguel Pinto dos Santos

Thesis to obtain the Master of Science Degree in

Mestrado Integrado em Engenharia Electrotécnica e de Computadores

Supervisor: Prof. Maria Margarida Campos da Silveira

Examination Committee

Chairperson: Prof. João Fernando Cardoso Silva Sequeira
Supervisor: Prof. Maria Margarida Campos da Silveira
Member of the Committee: Prof. Ana Luísa Nobre Fred

July 2021

Declaration

I declare that this document is an original work of my own authorship and that it fulfills all the requirements of the Code of Conduct and Good Practices of the Universidade de Lisboa.

Declaração

Declaro que o presente documento é um trabalho original da minha autoria e que cumpre todos os requisitos do Código de Conduta e Boas Práticas da Universidade de Lisboa.

Abstract

Alzheimer's Disease is a form of dementia which will become more present as life expectancy increases. Modelling the progression of this pathology would be beneficial for the medical community. The goal of this thesis is to correctly predict the diagnoses of patients, namely if they developed the disease or any stage that might lead to it.

Longitudinal datasets such as ADNI provided the cognitive tests, diagnostics, patient age and MRI data which were used as features for this work. RNN based models are indicated for this task as they are best suited for longitudinal data. To achieve this goal, in addition to a new RNN architecture termed Time-Aware Long Short Term Memory Network (T-LSTM), two other strategies were tested: time intervals (TI) between consultations, and generative models for the imputation of missing data. Varying TI are always present in some datasets due to acquisition protocols. Generative models achieve imputations based on the statistical distribution data.

The generative models achieved better results than the baseline imputation methods and part of the state of the art, despite not using as many features as desired. The T-LSTM allows for more data to be used than the LSTM network, creating bigger datasets which are advantageous for this task. However, its classification results are not better.

Concluding, the generative methods are competitive with the state of the art and might yield the best results in this task. The T-LSTM allows for more data to be used but its decay mechanism might be hindering its results.

Keywords

Alzheimer's Disease, RNN, T-LSTM, GAN, VAE.

Resumo

A Doença de Alzheimer é uma forma de demência que se prevê tornar mais presente à medida que a esperança de vida aumenta. Modelar o progresso desta doença seria benéfico para a comunidade médica. O objetivo desta dissertação é de prever com precisão o diagnóstico de vários pacientes, em específico se têm esta doença ou se estão em algum estado que possa progredir para tal.

Conjuntos de dados longitudinais como o ADNI fornecem dados sobre os pacientes. Neste trabalho foram usados testes cognitivos, diagnósticos clínicos, idades, e ressonâncias mágneticas. Os modelos de RNN são os mais indicados para lidar com estes dados. Para alcançar o objetivo desta dissertação, foram utilizadas várias estratégias, nomeadamente uma arquitetura de RNN nova (T-LSTM) e duas outras estratégias: o intervalo de tempo (TI) entre consultas, e modelos generativos para preencher dados em falta. Os TI não constantes estão sempre presentes neste conjunto de dados. Os modelos generativos preenchem os valores em falta com resultados derivados da distrubuição estatística dos dados.

Os dois modelos generativos obtiveram melhores resultados na classificação do que a baseline e parte do estado da arte, mesmo não utilizando tantos dados como desejado. A T-LSTM permite utilizar mais dados, algo vantajoso para esta tarefa. Os resultados na classificação da nova arquitetura não foram os melhores.

Concluindo, os modelos generativos são competitivos com o estado da arte e podem ter os melhores resultados nesta tarefa. A arquitetura proposta permite utilizar mais dados, mas o mecanismo de decaimento pode estar a obstruir os resultados.

Palavras Chave

Doença de Alzheimer, RNN, T-LSTM, GAN, VAE.

Contents

List of Figures

List of Tables

Acronyms

AD	Alzheimer's Disease
ADAS	Alzheimer's Disease Assessment Scale
ADNI	Alzheimer's Disease Neuroimaging Initiative
AUC	Area Under Curve
CN	Cognitively Normal
CSF	Cerebrospinal Fluid
DL	Deep Learning
DTI	Diffusion Tensor Imaging
EFPIA	European Federation of Pharmaceutical Industries and Associations
FDG	Fluorodeoxyglucose
GAIN	Generative Adversarial Imputation Network
GAN	Generative Adversarial Networks
GRU	Gated Recurrent Unit
GRU-D	Gated Recurrent Unit with Decay
GRUI	Gated Recurrent Unit for data Imputation
KNN	K-Nearest Neighbours
LSTM	Long Short Term Memory
MAR	Missing At Random
MCAR	Missing Completely At Random
MCI	Mild Cognitive Impairment
ML	Machine Learning
MMSE	Mini–Mental State Examination
MNAR	Missing Not At Random

MRI	Magnetic Resonance Imaging
MSE	Mean Squared Error
NACC	National Alzheimer's Coordinating Center
NN	Neural Network
PET	Positron Emission Tomography
RAVLT	Rey Auditory Verbal Learning Test
RNN	Recurrent Neural Network
ROC	Receiver Operating Characteristic
ROI	Region of Interest
TADPOLE	The Alzheimer's Disease Prediction Of Longitudinal Evolution
TI	Time Intervals
T-LSTM	Time-Aware Long Short Term Memory
VAE	Variational Autoencoders

1

Introduction and Motivation

Alzheimer's Disease (AD) is a neurodegenerative disease which slowly and progressively destroys brain cells and represents 60-65% of cases of dementia. First described by the German neurologist Aloïs Alzheimer, this disease affects cognitive function, presenting itself mainly with the loss of memory and, for example, through confusion and slower thinking. It happens through abnormal deposits of proteins forming amyloid plaques and tau tangles throughout the brain. With no currently known cure and as life expectancy increases with the advances in quality of life, the number of people affected with this pathology and other forms of dementia will most probably increase [8].

According to the European Federation of Pharmaceutical Industries and Associations (EFPIA), Europe has 10.5 million patients with some form of dementia, of which up to 80% suffer from AD. In the United States of America, AD is the sixth leading cause of death, with recent estimates placing it in third place. Modelling the progression of this pathology would be of great help for diagnosis, treatment purposes, caretakers and the national health services of each nation [9].

The typical first symptom of AD is more memory loss than what would be normal for the patient's age, but to an extent that does not interfere with normal day-to-day life. This is designated as Mild Cognitive Impairment (MCI). While not all patients who present MCI progress to AD, this stage precedes AD. After the onset of the disease itself, there are several stages, which, according to the National Institute of Aging [10], present themselves as follows:

- Mild AD: experiencing greater memory loss and other cognitive difficulties such as repeating questions, mishandling money and personality/mood changes, as well as struggling to complete normal tasks.
- Moderate AD: the brain is damaged in the parts which are responsible for the control of language, reasoning, sensory processing, and conscious thought.
- Severe AD: the plaques and tangles have spread throughout the entire brain, causing significant shrinking. Patients with severe Alzheimer's cannot communicate and are completely dependent on others for their care. Near the end, the person may be in bed most or all of the time as the body shuts down.

The growth of popularity of Machine Learning (ML) in recent years has been enhanced by widespread accessibility to cheap and powerful processors, graphics processing units, programming software like Python, ML libraries (like Keras and TensorFlow) and repositories. ML has thus become a powerful tool of great interest in research of multiple disciplines and industry, such as financial analysis, spam detection, image recognition and medical diagnosis and treatment.

Deep Learning (DL) is a specific field of ML which is characterised by a new take on learning representations from data that puts an emphasis on learning successive layers of increasingly meaningful representations. [1]. The ”Deep” on DL comes from layer depth, in the sense that there is more than one layer to the models. Adding more layers to the models comes at a cost of higher complexity and more difficult training.

To capitalise on these methods, complete datasets with many features and data are required, such as the ones provided by the Alzheimer's Disease Neuroimaging Initiative (ADNI) [7]. This initiative, which was launched more than a decade ago, unites researchers from all over the world with study data as they work to define the progression of AD. It has the goals of detecting AD at the earliest stage possible, developing biomarkers which will be used as predictors for cognitive decline, supporting advances in AD intervention and prevention, and treatment through the application of new diagnostic methods at the earliest possible stages, all while providing data to scientists in all of the world.

The datasets from ADNI are sequential data, meaning that the order of the data of each patient is important. Thus, Recurrent Neural Network (RNN), which were designed to process sequences by iterating through the sequence elements and maintaining a state containing information relative to what it has seen so far, are possible to use to make promising models to apply to datasets like the one aforementioned. These networks come at the cost of requiring large datasets and complete longitudinal data with equal time intervals between samples [1].

In this work, the models to be applied will take into account the handling of missing data and how to exploit time dependencies between observations with the goal of improving the current state of the art models for modelling progression of AD with RNN.

2

State of the Art

Contents

To model the progression of AD with RNN, it is necessary to have complete datasets. On top of this, the Time Intervals (TI) between the several consecutive sequences have to be constant, which is not true for some datasets like the ADNI dataset, if one were to use the whole dataset as provided. This happens due to missing data and/or the acquisition protocols for each kind of diagnosis of a patient [11].

2.1 Types of Missing Data

To first address data and how to treat it, it is necessary to know that, while datasets are not always complete, there are different ways how data can be missing from them. Following the definitions from A. R. T. Donders et al. [12], missing data can be separated into several types, like Missing Completely At Random (MCAR), where the missing data belongs to a random subset of the complete sample of subjects and the reason the data is missing is not related to any patient characteristics. An example of this would be, in the context of this work, an accidental loss of a patient's biomarkers for a certain consultation.

If there are observations missing because they depend on information that is not observed, they are called of Missing Not At Random (MNAR). Citing an example (from [12]), if "when asking a subject for their income there is a higher likelihood of not getting an answer when their level of income is higher". In this case, the missingness is not completely random but it is related to unobserved characteristics.

The case of data Missing At Random (MAR) happens when the missingness is related with the subject itself and not with factors exterior to the patient. It is said to be MAR because it is missing conditionally on an (un)observed characteristic of the patient.

When missing data falls into the categories of MAR and MCAR, most imputation techniques result in unbiased estimations of the missing values. On the other hand, if the missing data is MNAR, there is no universally agreed imputation method.

2.2 Missing Data Treatment

In the literature reviewed there were two ways to handle this problem: the model of RNN architecture allowed for the TI to not be constant, be it because the modules of the network used TI as an input parameter or because the TI was a feature itself, or the architecture used just the equally temporally spaced sequences as training data.

To tackle the problem of not having equally spaced longitudinal data for RNN, several authors have made progress in this area and have suggested ways to deal with the missing data problem. There are two main schools of thought in the literature presented, which are imputing the missing data or incorporating the missingness of data in the model and/or TI between the sequences.

2.2.1 Imputation

This approach is intuitive in the sense that there are several ways to perform operations with algorithms to fill in the missing data. As argued in Donders AR et al. [12] this does not necessarily produce good results for medical data because most of these approaches are usually focused on a single variable rather than on the totality or groups of variables.

One of the most commonly found imputations in literature is the Forward Imputation [13], [14], [3], where the missing value is replaced with the last known value in the data series or, in case the first value is missing, Z. C. Lipton et al. [14] proposed the replacement with the median of the variable before any other imputations.

A similar method to the previous one is the Imputation through Linear Interpolation [15], [16]. It is made with the linear interpolation of the two closest neighbouring points of the missing data, within subject, such that the missing points take the value of the interpolation.

Mean, Mode or Median Imputations are also widely used [17], [18], [19] and work by replacing the missing values with the mean, mode or median of each variable, respectively. These imputations can be done in respect to the values of each variable of either individual patients or all patients. Usually these three methods were also coupled with RNN models which took into consideration the TI, which can be done using the TI as features, or with modified RNN units, as will be later discussed in this chapter.

Model Filling, where the proposed model is used as a way to generate missing data, was also applied and has a superior performance than that of Linear Regressions as shown by H. Kim et al. [20]. This method works by using RNN to predict the missing data of a subject based on previous observations or examinations and it can also be combined with a Linear Interpolation as seen in the work of M. Nguyen et al. [13]

Weighted K-Nearest Neighbours (KNN) is yet another form of imputing missing data which takes into account the other visits of the patient and chooses the K nearest candidates (i.e. the candidates with the least distance) and then imputes with the mean values of the K nearest candidate values. When

combined with Interpolation, this method outperformed the baseline in result quality and took much less time to run, as shown by S. Daberdaku et al. [15].

Another method of handling missing data is simply replacing all the missing points by zero. This method, called Zero Imputation, appears to work better with Long Short Term Memory (LSTM) based on the results of Z. C. Lipton et al. [14]. The authors suggested that the LSTM may be learning to recognise missing values implicitly by recognising a tight range about the value zero and inferring that this is a missing value. The authors also suggests that the LSTM may be learning to recognise missing values implicitly and that imputations might interfere with this learning ability.

More complex state-of-the-art methods are through the usage of Generative Adversarial Networks (GAN) [21] adapted for longitudinal data [4], [5] or Variational Autoencoders (VAE) adapted for this same scenario. Both these methods try to learn the underlying distribution of the data, rather than imputing feature by feature, and use that as an advantage against the previously introduced methods.

GAN are made up of a generator and a discriminator. The generator is trained to maximise the discriminator's misclassification rate. The discriminator is trained to minimise the classification loss. The goal with the Generative Adversarial Imputation Network (GAIN) adaptation is to make the generator learn how to create longitudinal data that the discriminator cannot distinguish from real data. This way, the generated data can be used to impute the missing data.

VAE are a type of Autoencoder which encode data that will be regularised into a space of lower dimension than that of the input. This space is called latent space and it will be following a probabilistic distribution before being decoded. The goal is to generate longitudinal data from decoding different points in that latent space that will resemble the distribution of the data [22], [23]. The data generated for an instant with missing data can be used to perform imputation of the and has been used previously in different kinds of datasets [24], [25] that were not related to AD.

2.2.2 Incorporating Missingness of Data

In terms of incorporating the missingness of data and/or TI into the models, this can be done by changing the RNN architectures or by using TI as features. Nonetheless, there is data treatment necessary to implement these methods.

To incorporate the missingness of data, it is necessary to label where data is missing for the features. In the case of certain features being composed exclusively of missing data, they are to be excluded from the cost function. This procedure is named Masking and is widely used [18], [19].

To implement TI, there is a need for arrays (one per patient) derived from the dataset where each of the array entries correspond to a TI between two consecutive consultations. If one were to incorporate the TI and impute the missing data, these arrays would contain the time between every consecutive consultation, regardless whether the points are missing or not. This is because the missing points would be later imputed. However, if incorporating the missingness of data, the TI would be calculated between non-missing data points, as the formulation in the following figure exemplifies

$$X = \begin{bmatrix} 1 & 2 & NA & NA & 0 & 6 \\ 0 & NA & 0 & 3 & NA & 8 \end{bmatrix} \qquad \Delta_{t1} = \begin{bmatrix} 0.0 & 0.2 & 0.1 & 0.2 & 1.2 & 0.9 \\ 0.0 & 0.2 & 0.1 & 0.2 & 1.2 & 0.9 \end{bmatrix}$$

$$t = \begin{bmatrix} 0.0 & 0.2 & 0.3 & 0.5 & 1.7 & 2.6 \end{bmatrix} \qquad \Delta_{t2} = \begin{bmatrix} 0.0 & 0.2 & - & - & 1.5 & 0.9 \\ 0.0 & - & 0.3 & 0.2 & - & 2.1 \end{bmatrix}$$

Figure 2.1: Computation of TI since last observation.

Notice that values of *X* filled with *NA* correspond to the missing data. Each row of *X* (and consequently each row of each Δ_t matrix) corresponds to a different variable and, for the purpose of understanding the TI computation, each row can be examined independently from each other. Furthermore, the timestamps in array *t* apply to each row of *X*.

The entries of Δ_{t1} are filled with the TI between each two successive points of *X*, calculated with the difference between the respective timestamps of vector *t*. This matrix was calculated for a situation where the missing data will be later imputed. The first column will always have the value of 0 as it corresponds to the starting point.

In the case of Δ_{t2}, the values of the TI were only calculated between each two consecutive points where data is not missing. To exemplify, the fifth entry in the first row is calculated by subtracting the values of t in the fifth and second instances (as this is the nearest previous instance of non-missing data), and thus the obtained value is calculated as $1.7 - 0.2 = 1.5$. This matrix corresponds to the situation where the missingness of data will be incorporated.

Finally, to give an overview of the different methods the following table is presented.

Table 2.1: Brief Description of the diverse methods of handling missing data found in literature.

Method	Brief Description	Type
Forward Imputation [13], [14], [3]	Missing value replaced with last known value	Imputation
Linear Interpolation [15], [16]	Missing value replaced with interpolation of two closest neighbours	Imputation
Mean, Mode, Median Imputation [17], [18], [19]	Missing value replaced with mean, mode, median value (respectively)	Imputation
Model Filling [20], [13]	Missing value replaced with output of the proposed model	Imputation
Weighted KNN [15]	Missing value replaced by the mean of KNN	Imputation
Zero Imputation [14]	Missing value replaced with zero	Imputation
GAN [4], [5]	Missing value replaced with generated data which mimics the real data	Imputation
VAE [23], [24], [25]	Missing value replaced with generated data from a decoding of latent space	Imputation
TI as Feature [17]	TI used as a feature in the proposed model	Missingness of Data
Masking [18], [19]	Missing data is not used	Missingness of Data

2.3 RNN Architectures

Sequential data in general is typically handled by RNN or 1D Convnets, as these networks are specifically designed to handle them and have becoming increasingly popular. When compared to other networks such as, for example, the Multilayer Perceptron (MLP), the RNN outperform them as they have what is called Parameter Sharing (or Weight Sharing). This means that the weights are shared across the layers of the network, which is advantageous as often sequences have slightly different orders but the same meaning.

From the definition of F. Chollet [1], RNN process sequences "by iterating through the sequence elements and maintaining a state containing information relative to what it has seen so far" [1]. They are a type of Neural Network (NN) with an internal loop. The following figure shows a simple diagram of the RNN

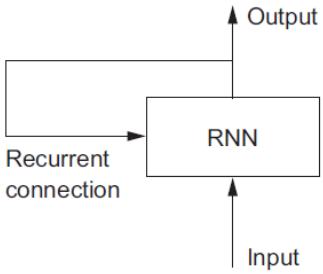

Figure 2.2: RNN scheme taken from [1].

The RNN state is reset between processing two different sequences so that each one is perceived as an input to the network. This way the network internally loops over elements of the sequence [1].

Introduced by Hochreiter & Schmidhuber in 1997 [26], the LSTM is a variation of the simple RNN aiming to deal with the vanishing gradient problem (i.e. when the gradient tends to zero when training the network with back-propagation). This architecture solves this problem by having the capacity to carry information across timesteps by using an additional data flow that carries information across timesteps (called Carry) [1]. The following figure depicts the structure of the LSTM

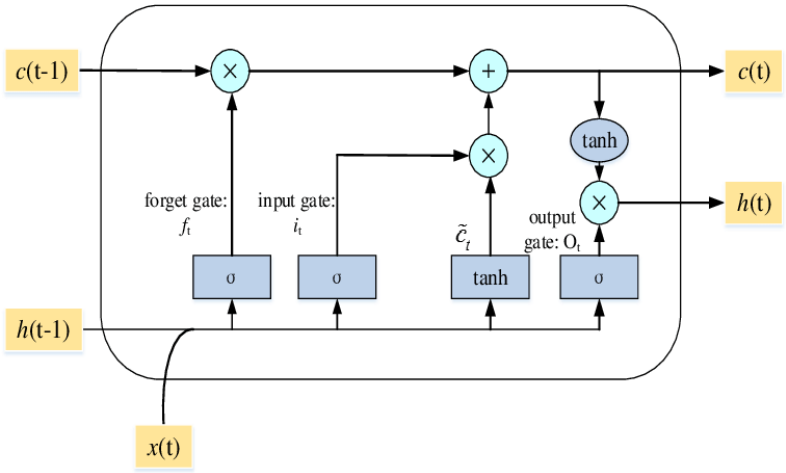

Figure 2.3: LSTM, taken from from [2].

Due to its structure, there are several gates with different goals in this architecture. The forget, input and output gates have the main goal of selecting which information to preserve. These three gates are calculated as follows

$$f_t = \sigma(W_{fx}x_t + W_{fh}h(t-1) + b_f), \tag{2.1}$$

$$i_t = \sigma(W_{ix}x_t + W_{ih}h(t-1) + b_i), \tag{2.2}$$

$$o_t = \sigma(W_{ox}x_t + W_{oh}h(t-1) + b_o), \tag{2.3}$$

where σ are nonlinear activation functions (such as the sigmoid or hyperbolic functions, for example), W are the weights of the cell and b the biases.

The intermediate state is given by

$$\tilde{C_t} = tanh(W_{cx}x_t + W_{ch}h(t-1) + b_c). \tag{2.4}$$

Notice that the activation is always an hyperbolic tangent (commonly referred to as $tanh$) and that the intermediate state also has weights and a bias.

Finally the memory cell and hidden state are updated, respectively, as

$$C(t) = f_t \odot C(t-1) + i_t \odot \tilde{C_t}, \tag{2.5}$$

$$h(t) = O_t \odot tanh(C_t). \tag{2.6}$$

The $\odot$ symbol denotes the Hadamard product (entry-wise multiplication for matrices).

In the literature, on top of the different ways to handle data, there are also some modifications of each network. One of them is the Time-Aware Long Short Term Memory (T-LSTM) [3], which was designed to handle varying TI between sequences. In this network the TI between the sequences are used as an input to create a Discounted Short-Term Memory which has the goal of creating an Adjusted Previous Memory.

In the figure below, the architecture of the T-LSTM is shown

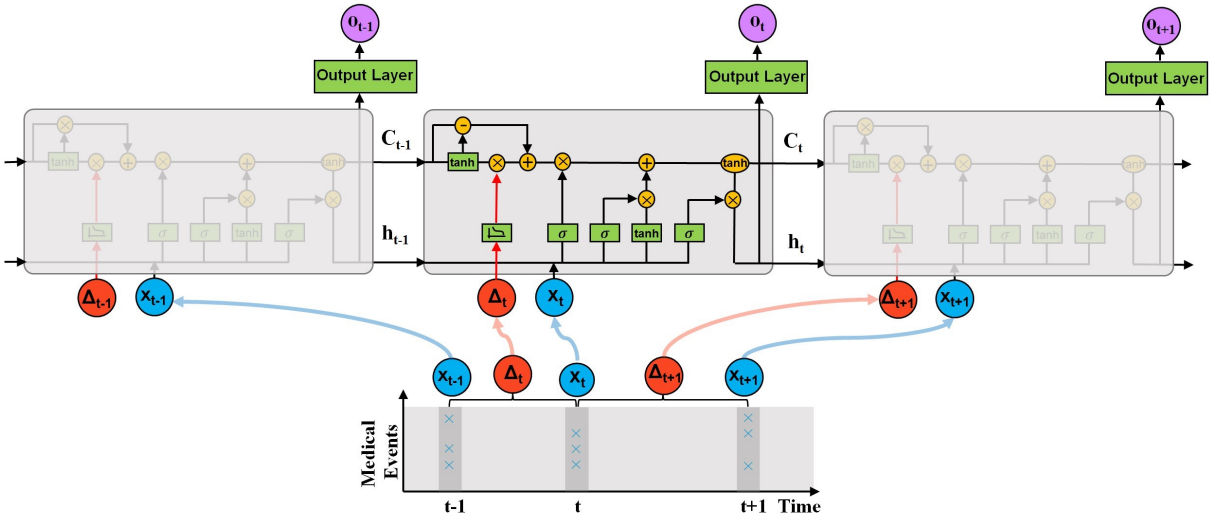

Figure 2.4: T-LSTM architecture from I. M. Baytas et al. [3]

To define this Adjusted Previous Memory (C^*_{t-1}), there are several steps, all defined by I. M. Baytas et al. [3]. Firstly, the definition of the Short-Term Memory is as follows

$$C^S_{t-1} = tanh(W_d \cdot C_{t-1} + b_d). \tag{2.7}$$

Followed by the definition of the Discounted Short-Term Memory

$$\hat{C}^S_{t-1} = C^S_{t-1} \cdot g(\Delta_t), \tag{2.8}$$

which results of the application of the nonlinear function g, that has the goal to introduce a penalty which increases with the value of a TI. An example of such, and the implemented function by default in the T-LSTM architecture, is the following

$$g(\Delta_t) = \frac{1}{ln(e + \Delta_t)}. \tag{2.9}$$

The Long-Term Memory is defined as the difference between the Current Memory and the Short-Term Memory

$$C^T_{t-1} = C_{t-1} - C^S_{t-1}. \tag{2.10}$$

Finally, the Adjusted Previous Memory is given by

$$C^*_{t-1} = C^T_{t-1} + \hat{C}_{t-1}. \tag{2.11}$$

This last expression can be rewritten taking in consideration the definitions of Long-Term Memory (2.10)

and of Discounted Short-Term Memory (2.8)

$$C^*_{t-1} = C_{t-1} - C^s_{t-1} + C^S_{t-1} \cdot g(\Delta_t) = C_{t-1} - C^S_{t-1}(1 - g(\Delta_t). \tag{2.12}$$

With this formulation, it is highlighted that the contribution of the Short-Term Memory depends of the value of $g(\Delta_t)$. Moreover, if this function g takes value 1 (which would happen with $\Delta_t = 0$ in the formulation above), the Short-Term Memory does not contribute to the Adjusted Previous Memory and thus the latter would be defined as solely the Current Memory (like in a standard LSTM cell as seen in Equation (2.4)). For large values of Δ_t, the function g tends to 0 and the Short-Term Memory discounts more as it is attributed a higher weight in the expression above. In the limit situation where $g = 0$, the Adjusted Previous Memory would have the exact same formulation that of the Long-Term Memory defined previously in Equation (2.10).

The literature has other architectures such as the Gated Recurrent Unit (GRU) [27], which is similar to the LSTM, but with less parameters. There's also a modification found in literature [18], termed Gated Recurrent Unit with Decay (GRU-D), that aims to add a trainable decay γ under the assumption that the missing variable tends to be close to some default value due to the homeostasis mechanism of the human body. The trainable decay follows an exponential function with a negative exponent. To further illustrate these architectures, the following figure taken from Z. Che et al. [18] compares the regular GRU with the GRU-D, respectively next to each other.

Figure 2.5: GRU (left) and GRU-D (right) side by side. In blue the differences between both modules are highlighted.

Notice that the GRU-D is being used with Masking, hence the reference to a mask in the figure above.

As the elements of the GAIN in Y.Luo et al. [4], the authors propose a module similar to the GRU-D, which is called Gated Recurrent Unit for data Imputation (GRUI). The module is presented next to the regular GRU in the following figure, taken from the aforementioned paper, where β_{t_i} is a parameter to be learned and δ is the elapsed time between measurements

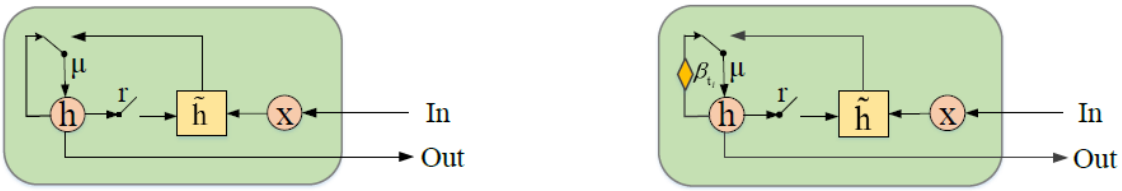

Figure 2.6: GRU (left) and GRUI (right) side by side. In yellow is the difference between the two networks, which is a time decay vector (β).

Tthe difference between the GRU-D and GRUI is that the former has a trainable decay at the input while the later does not, while they share a trainable decay before the output gate.

Similarly to the GRU-D, this decay is an exponential with a negative power, defined as follows for this case (and with analogous variables for the GRU-D)

$$\beta_{t_i} = \frac{1}{e^{max(0, W_\beta \delta_{t_i} + b_\beta)}}. \tag{2.13}$$

Notice that the values of W_β and b_β are to be learned.

2.4 Methods Applied to AD

The methods referred in the previous section have been applied to AD sequential data or just medical sequential data. All architectures have been applied in the same conditions. In Table 2.2 there is a list of what was tested, by whom, the number of subjects and to which results did it lead for the AD sequential data. Table 2.3 is similar to the previous one with the main difference that it uses RNN for other kinds of medical sequential data.

As shown by the work of M. Nguyen et al. [13], Model Imputation was outperformed by Forward Imputation.

The two most commonly used methods were the Forward Imputation and Masking. Masking performed worse than Forward Imputation or Model Imputation when comparing the results of M. Nguyen et al. [13] and M. M. Ghazi et al. [19]. Zero Imputation, however, performed slightly worse than Masking and slightly better than Forward Imputation [19].

In Table 2.3 it is possible to draw a comparison between the GAN and Masking, as they used simi-

lar variations of the GRU for the same dataset despite having a much different number of patients. In this case the GAN outperformed Masking.

Having only access to the Mean Squared Error (MSE) and not to accuracy or other metrics, the T-LSTM showed promising results which seem very relevant for the context of this work but are hard to compare to the other works. The best performance in Table 2.3 also happens to be the only one which used the National Alzheimer's Coordinating Center (NACC) dataset, with no other usage of the same dataset possible to compare to. The two biggest particularities in the work of T.Wang et al. [17] were the imputation being done with Mean, Mode and Median Imputations and the usage of TI as a feature in the model with the biggest number of patients out of all AD related datasets.

Other non RNN methods were employed for this task, as referenced in Table 2.3. Both of them mainly used the images of the Magnetic Resonance Imaging (MRI) and achieved results which were close to the RNN results for the The Alzheimer's Disease Prediction Of Longitudinal Evolution (TADPOLE) datasets, despite using a lower amount of patients and using only full data (thus not making use of any imputation methods).

Table 2.2: Summary of the state of the art of RNN in AD. The accuracy column is regarding to each one of the Missing Data Treatment entries, respectively (in the cases where there's more than one pre-processing in comparison). (1) - Value respecting to normal cognitive functions vs MCI vs AD. (5) - only has MSE and some other statistics where the T-LSTM outperforms the regular LSTM.

Authors	Features or Dataset	RNN Architecture	Missing Data Treatment	Patients	Accuracy(%)
M. Nguyen et al., 2018 [13]	TADPOLE	LSTM	Forward Imp., Model Imp.	1700	86, 83
T. Wang et al., 2018 [17]	TI, NACC	LSTM	Mean, Median & Mode Imp.	5432	99.06
M. M. Ghazi et al., 2019 [19]	TADPOLE	LSTM	Masking	1737	75.96(1)

Table 2.3: Summary of the state of the art on either other longitudinal datasets or with non RNN methods. The accuracy column is regarding to each one of the Missing Data Treatment entries, respectively (in the cases where there's more than one pre-processing in comparison). (1) - Paediatric Intensive Care Unit. (2) - Mortality Rate. (3) - Parkinson's Disease.

Authors	Features or Dataset	RNN Architecture	Missing Data Treatment	Patients	Accuracy(%)
Z. C. Lipton et al., 2016 [14]	PICU(1)	LSTM	Forward Imp., Zero Imp., Masking	10401	86.00, 86.62, 87.30
Z. Che et al., 2016 [18]	TI, MIMIC-III, PhysioNet(2)	GRU-D	Masking	8000 & 19714	71.23, 83.70
I. M. Bayas et al., 2017 [3]	PPMI(3)	T-LSTM	Forward Imp.	654	- (5)
Y. Luo et al., 2018 [4]	PhysioNet(2)	GRUI	GAN	4000	86.03
D. S. Cohen et al., 2019 [28]	ADNI	ANN, 1D CNN	Not Used	800	87.20, 88.28
J. Neelaveni et al., 2020 [29]	ADNI	SVM, Decision Tree	Not Used	-	85, 83

3

Proposed Methods

Contents

Keeping in mind the goal of modelling the progression of AD with RNN, the proposed methods will have to handle the missing data in the dataset with the goal of meeting the needs of the proposed RNN architectures. The first section of this chapter is about the assumptions taken about the missing data and its treatment, the second section is about the RNN based models proposed for the classification task of this work and the third section is about the proposed methods for classification as well as constituting a baseline.

3.1 Missing Data Treatment

For the scope of this work, missing data was considered MCAR, meaning that the data which was missing was completely random and unrelated to any patient characteristics. There are also situations where the data was MNAR, namely when a consultation was not acquired due to acquisition protocols, which will be highlighted in the following chapter. In practice, both types of missing data underwent imputation in the same manner. The dataset consisted of n features, with t consultations for each of the p patients.

The TI are not constant between the ADNI clinical data, so the procedure was to acquire two sets of data: one with constant TI which were always 12 months apart between consultations, and another one that had no restrictions regarding the elapsed time between consultations. Both datasets have five consultations per patient, with the condition that every single consultation has a diagnosis. Thus, the first set of data was evenly spaced in time while the second was not.

This way, the LSTM network was used to constitute models for the generative imputation methods and for classification, which were trained and tested with the first set. The T-LSTM had high computational requirements and a complex TensorFlow 2.0 implementation. The latter caused incompatibilities with the Keras based implementations of the generative models. Consequently, the second dataset only underwent the simple baseline imputation techniques and was only used to train and test the T-LSTM classification model.

Imputations should aim to work based on the general frame of the features of each patient rather than inferring about the missing values of a feature using just the existing data of that feature itself [12]. Thus, the strategies which were expected to produce better results to handle missing data, and consequently the ones chosen for this work, were two generative models. They were the GAN network, more specifically the model introduced by Y. Luo et al. [4] (termed GAIN), and the VAE [22], more specifically the β-VAE [30]. Both GAN and VAE based models have proven themselves for imputation of longitudinal data in literature [4], [5], [24], [25] but these powerful methods had not yet been tested for modelling AD

progression with longitudinal data.

3.1.1 GAIN

The GAIN, much like the GAN, is made up of a generator (G) and a discriminator (D). The G is trained to maximise the D's misclassification rate while the D is trained to classify which data is real or generated. The method implemented by J. Yoon et al. [5] is a variation of the GAN which aims to impute missing data by using the learned distribution of the existing data. The following figure shows a standard GAN for longitudinal data

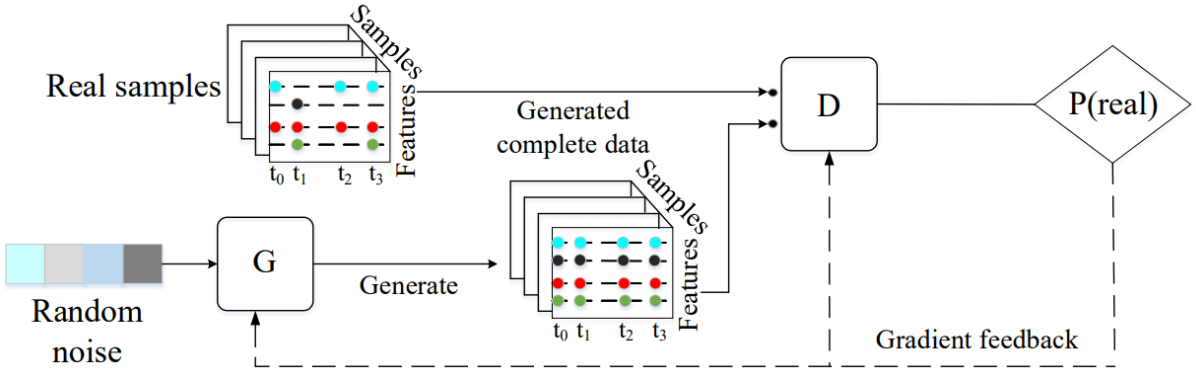

Figure 3.1: GAN for longitudinal data taken from [4].

The goal with the GAIN adaptation is to make the generator learn how to impute longitudinal data such that the discriminator cannot tell which data is real and which one is imputed, even when given a hint vector that suggests some points in the data are imputed. To implement this variation of the GAN, first, it will be required to compute mask arrays M, where each entry indicates if a value exists (taking value of 1) or not (taking value of 0) for each variable.

In a following step, the GAIN model will have to learn the distribution of the original dataset. To achieve this, the G will have to learn a mapping which tries to map this random noise vector into a realistic time series, where Z is a random noise vector, $\tilde{X}$ the original data at the input of the G, M the masking vector and $\bar{X}$ the imputed data.

$$\bar{X} = G(\tilde{X}, M, (1 - M) \odot Z), \tag{3.1}$$

$$\hat{X} = M \odot \tilde{X} + (1 - M) \odot \bar{X}. \tag{3.2}$$

The D will contain both the incomplete real samples of data and the completed fake data generated by the G as well as a hint vector as the following figure depicts

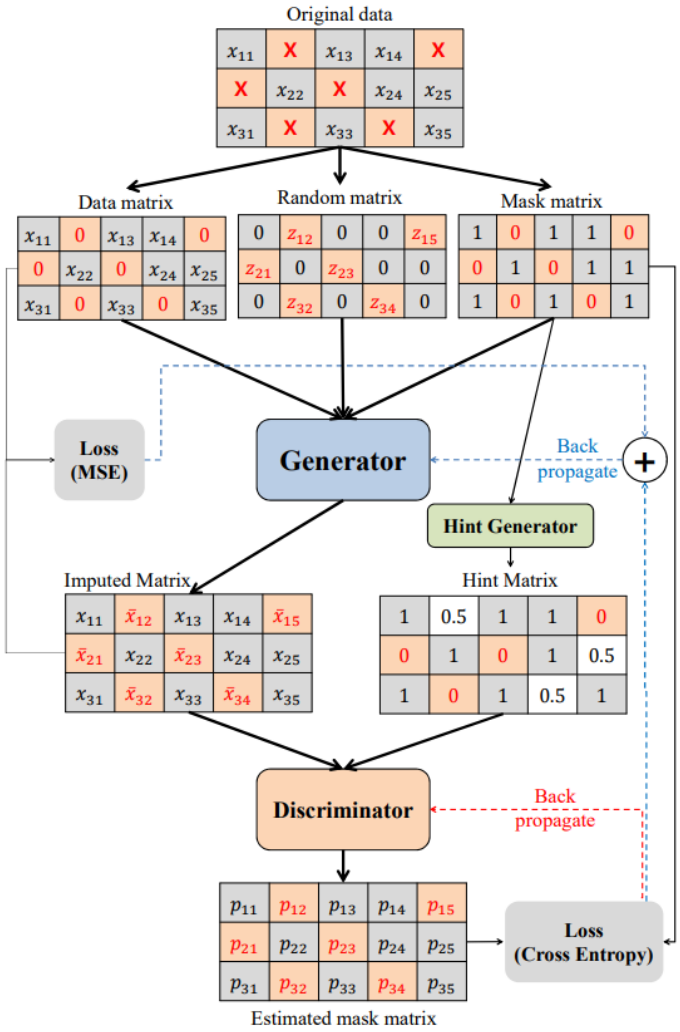

Figure 3.2: GAIN architecture, taken from [5]. In the Original data, X marks missing data, which is replaced with 0 when it is input in the generator.

As with any typical GAN, this variation is also trained adversarially. The training of the standard GAN would have the D maximise the probability of distinguishing between fake and real data, adversarial to the G minimising the probability of D distinguishing with success. The twist for the GAIN is that it is trained adversarially in respect to the mask, that is, D is trained to maximise the probability of predicting M while G is trained to minimise the probability of D successfully predicting M [5]. Thus, the formulation of the generator and the objective of the GAIN, taken from [5] are as follows

$$\min_{G} \max_{D} V(D,G) = \mathbb{E}_{\hat{X},M,H}[M^T log D(\hat{X},H) + (1-M)^T log(1-D(\hat{X},h))]. \tag{3.3}$$

To highlight that this function is indeed in respect to the masking vector M, the following definition of the

loss function (based on the cross-entropy) is defined by Y. Yoon et. al.

$$\mathbb{L}(a,b) = \sum_{i=1}^{d} [a_i \, log(b_i) + (1 - a_i) log(1 - b_i)]. \tag{3.4}$$

Considering that the discriminator outputs an estimated masking vector $\hat{M}$ as seen in Fig. 3.2

$$\hat{M} = D(\hat{X}, H). \tag{3.5}$$

When combining these last two expressions, the outcome of rewriting (3.3) highlights that the training is a function of the masking vector

$$\min_{G} \max_{D} V(D,G) = \mathbb{E}[\mathbb{L}(M, \hat{M})]. \tag{3.6}$$

The next step is to find the best vector z for any incomplete time series x such that $G(z)$ is similar to x. To achieve this, the discriminator outputs a mask matrix which will be compared with the real mask matrix by computing its cross-entropy and used to train the D. This loss, alongside with the MSE of the non masked points of the imputed matrix and of the data matrix, will be summed and used to train the G.

A new feature in this GAIN network is the hint mechanism. The hints are a subspace of the masked values. The discriminator is allowed to try and tell apart values that otherwise it would not be able to access due to the mask. This mechanism exists, according to the authors [5], as a way to prevent cases where there are several distributions that G could reproduce that would be optimal in respect to D but that would not be optimal for the imputation problem.

Training these networks can be quite challenging due to the mode collapse problem, which happens due to the fact that the optimal generator, for a fixed discriminator, is a sum of deltas on the points the discriminator assigns the highest values [21].

3.1.2 β-VAE

VAE [22] [23] are a generative model consisting of an Autoencoder network and a random sampling layer. Autoencoders are networks with two parts: an encoder and a decoder. The encoder uses the data as input and, as output, it has a representation of the input data in reduced dimensionality, represented in what is called of latent space. The decoder takes this representation in the latent space as input and decodes it as output, thus creating a reconstruction, as close as possible, of the original data.

Autoencoders and VAE work for any kind of data and longitudinal data is no exception. They are thus designed to encode a representation of the input in a lower dimension space and then do the closest reconstruction possible in respect to the original data. Typically, an Autoencoder is trained in respect to reconstruction loss, using the MSE or cross-entropy as loss function [31].

The random sampling layer works by taking the output of the encoder and transforming it in two vectors, each one with the dimension of the latent space: σ for the standard deviation and μ for the mean of each variable. With these two vectors, the latent space becomes continuous and will follow a probability distribution due to some changes in how the loss will be calculated. Each encoding is now a point with an "area" given by its standard deviation, centred at the mean, in the latent space. Thus, not just the point centred at the mean is of its class but also the cluster in the surrounding "area".

The typical VAE can have its architecture represented as follows

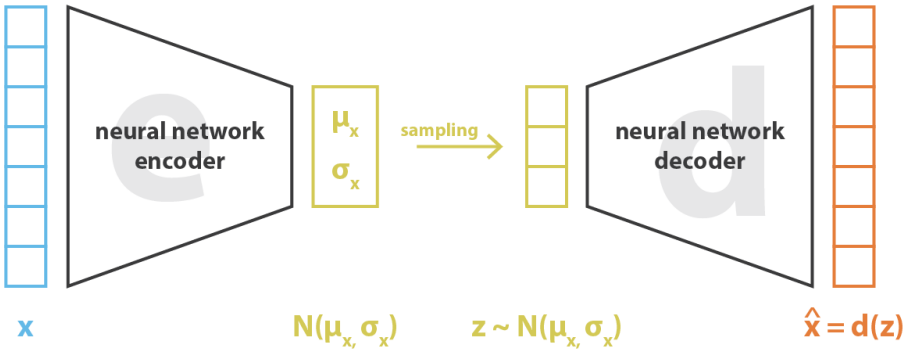

Figure 3.3: VAE architecture, taken from [6].

This random sampling changes the loss function into two components: one part for the reconstruction loss and another one for the Kullback-Leibler divergence (or KL loss, in the VAE context) of the distribution in the latent space. The Kullback-Leibler Divergence is, intuitively, how much two probability distributions diverge from each other. A value of 0 means that the two distributions are the same (no divergence) and the higher this value the more divergent are the two distributions. Its formal definition [32] is as follows

$$D_{KL}(p_1||p_2) = \sum_{x \in X} p_1(x) log \frac{p_1(x)}{p_2(x)}. \tag{3.7}$$

In the VAE context, these loss functions are defined as follows, for inputs of dimension p, latent space of

dimension n, original data X_i and decoded data $\tilde{X}_i$:

$$\text{Reconstruction Loss} = \frac{1}{n}\sum_{i=1}^{p}(X_i - \tilde{X}_i), \tag{3.8}$$

$$\text{KL Loss} = \sum_{i=1}^{n}(\sigma_i^2 + \mu_i^2 - log(\sigma_i) - 1), \tag{3.9}$$

$$\text{Loss} = \text{Reconstruction Loss} + \text{KL Loss} = \frac{1}{n}\sum_{i=1}^{p}(X_i - \tilde{X}_i) + \sum_{i=1}^{n}(\sigma_i^2 + \mu_i^2 - log(\sigma_i) - 1). \tag{3.10}$$

Notice that this KL Loss formulation is specific of the VAE. In order to assign a different weight to the KL Loss, as to control how much this term affects the overall loss and the distribution of the latent space, the following formulation of the loss is used

$$\text{Loss} = \frac{1}{n}\sum_{i=1}^{p}(X_i - \tilde{X}_i) + \beta\sum_{i=1}^{n}(\sigma_i^2 + \mu_i^2 - log(\sigma_i) - 1), \tag{3.11}$$

and this changes, in literature, the name of this generative model from VAE to $\beta-$VAE [30]. In practice, changing β to 0 transforms the VAE in a regular Autoencoder. A very large value of β forces the latent space of the VAE to be centred at the origin and with little variance (i.e. the "area" around the origin is very small), whereas a very small value of β gives freedom to the latent space to take whichever shape the encoding process will produce.

Lastly, the following figure is a good intuition of the purpose of the latent space through 2D images

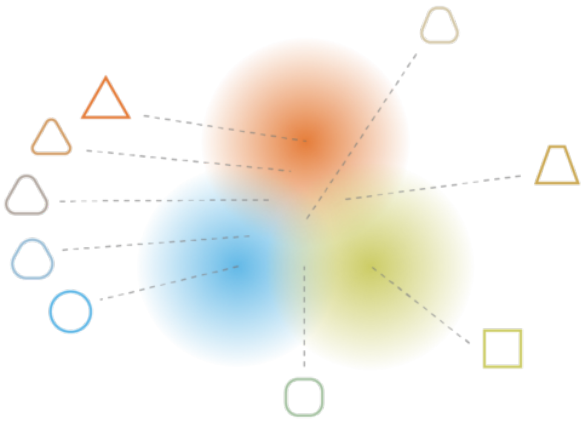

Figure 3.4: Latent Space of a VAE architecture trained for images (of circles, squares and triangles), taken from [6].

The latent space being continuous allows for the sampling of these shapes which are similar to the

training set of circles, squares and triangles but with some changes. This same principle can be applied to the data of this work which, instead of geometric figures, uses the data of patients who may be any of the three classes of Cognitively Normal (CN), MCI and AD.

3.2 RNN Architecture for Classification

The T-LSTM has the advantage of dealing with data which is not separated by constant TI, as discussed in Chapter 2. It has shown to be promising [3], which leads to a hypothesis regarding how good this network could be at modelling the progression of AD with this dataset. The results of the model made with the T-LSTM will be compared with the results of the LSTM based model results and, for classification purposes, the number of layers, units, dropout and any other hyperparameters of both networks was determined by taking into account the state of the art.

3.3 Proposed Method for Classification

With the purpose of constituting a baseline, Masking, Mean and Forward Imputations are the three proposed methods as they achieve state of the art results, are simpler than the GAIN or the β-VAE, are widely used [13], [14], [3], [18], [19] and can be applied to equality to the unevenly and evenly spaced datasets.

Masking required to fill in the missing values with a mask value. The Mean Imputation replaced the missing values with the mean of the known values and the Forward Imputation worked by replacing the missing values with the previously known value. Each one of the two datasets discussed in this chapter was firstly imputed with each technique and then saved. The only exception were the full datasets, which were obtained by dropping all patients with missing data. In a following step, the previously obtained datasets are imputed and then used as input to train and then test their respective classification RNN model, which produced one classification per consultation (between CN, MCI or AD), thus producing five classifications in total per each patient.

4

Dataset

Contents

The components and acquisition protocols of the original dataset, as well as which features were used, their missing rates and the size and breakdown of the resulting datasets will be discussed in this chapter.

4.1 Dataset

The following figure shows that, initially, there's an interval of three months between the baseline and the next visit but, for example, month 12 and month 18 are six months apart

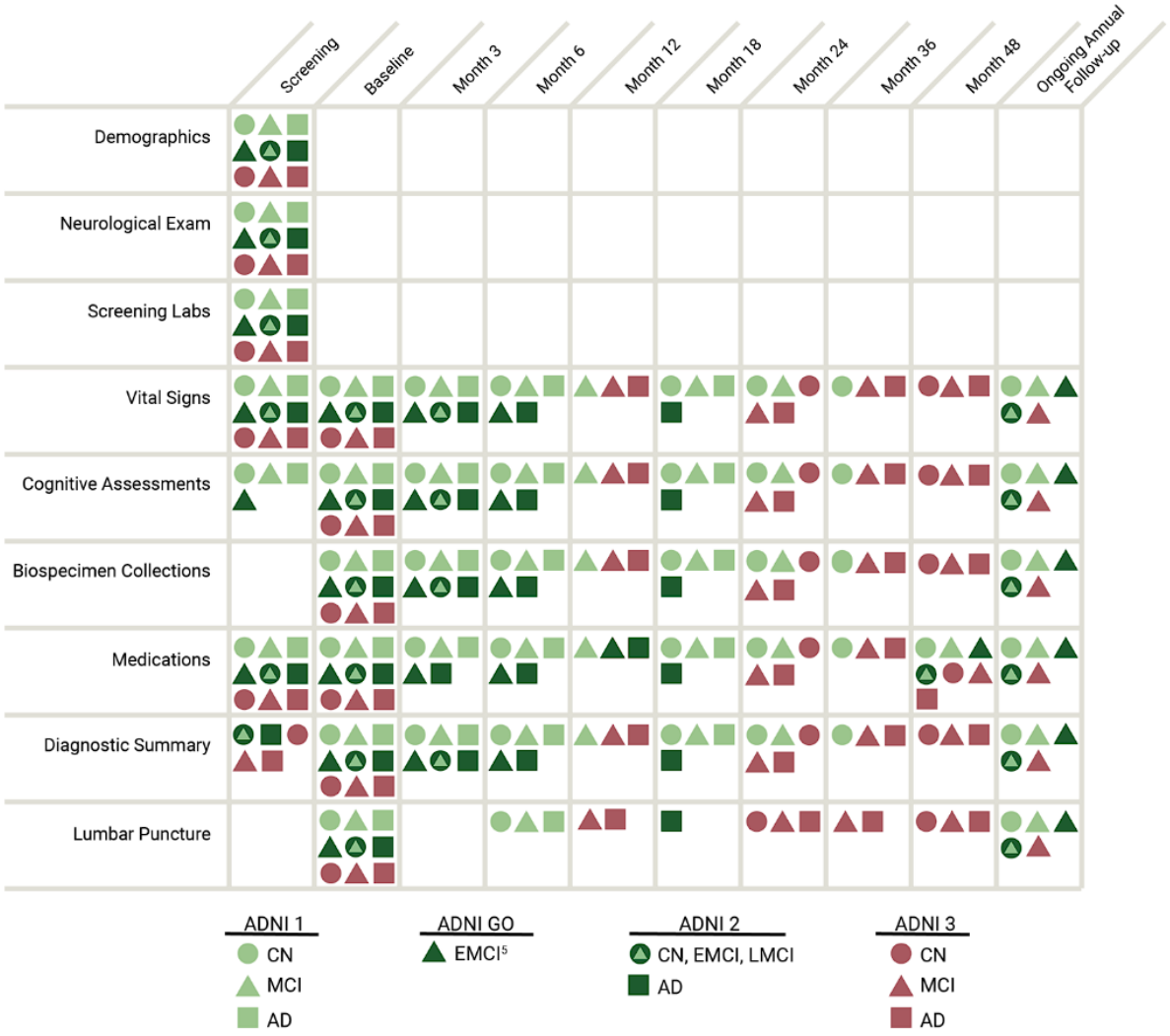

Figure 4.1: Composition of the ADNI dataset, as taken from [7].

Another relevant protocol of data acquisition is that of the MRI images, which is as follows

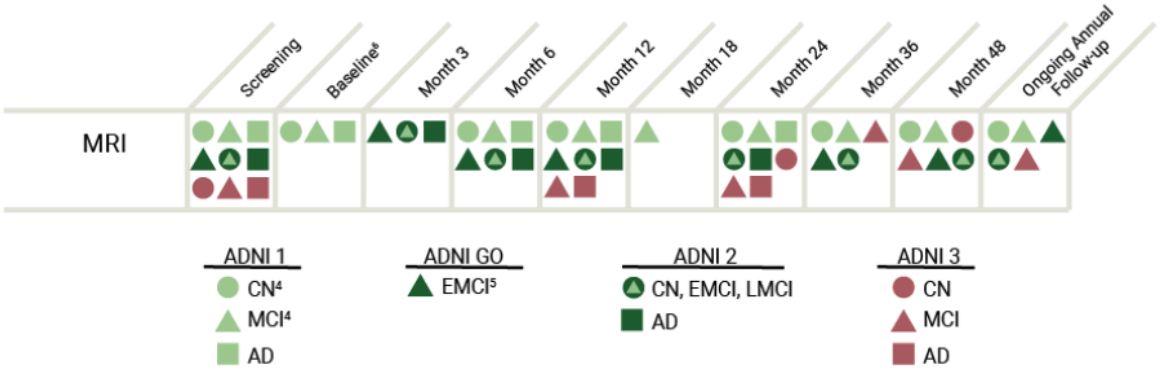

Figure 4.2: MRI images in the ADNI dataset, as taken from [7].

TADPOLE dataset has several biomarkers and screening measures [33], namely

- Cognitive tests (such as the Mini–Mental State Examination (MMSE), the Rey Auditory Verbal Learning Test (RAVLT) and the Alzheimer's Disease Assessment Scale (ADAS)).
- MRI Region of Interest (ROI)s which measure brain structural integrity.
- Fluorodeoxyglucose (FDG) ROIs which measure cell metabolism (where cells affected by AD show reduced metabolism).
- AV45 Positron Emission Tomography (PET) ROI averages which measure the amyloid-beta load in the brain (which is a protein that misfolds and then leads to AD)
- AV1451 PET ROIs which is similar to the previous one but for the tau protein
- Diffusion Tensor Imaging (DTI) ROI measures of micro-structural parameters related to cells and axons
- Cerebrospinal Fluid (CSF) biomarkers of the amyloid and tau levels in the CSF
- Others, such as the demographic information like age, gender, education and the diagnosis (CN, MCI and AD) of the patients.

4.2 Dataset Processing

To form the datasets which were used in this work, there was a selection of features, removal of patients with a number of consultations different from five and the removal of all features with no data.

The MRI data, which has its acquisition protocol is in Fig.(4.2), the cognitive tests, the diagnostics and the patient ages, which have their acquisition protocols respectively in the "Cognitive Assessments", "Diagnostic Summary" and "Demographics" rows of Fig.(4.1), were used in this work. This was due to the PET data only having measurements for the first consultation of most patients, albeit incomplete sometimes. The same issue happened with the CSF data. As these measurements are more recent, the DTI data had really high missing rates due to many ancient patients not having any data for this biomarker.

The MRI ROI data provides information regarding brain volumetry in the areas most affected by AD, namely the Amygdala, Entorhinal Cortex, Hippocampus, Temporal Lobe, Parietal Lobe, Frontal Lobe, Ventricles, Fusiform Gyrus, Cingulate Gyrus and Precuneus [10], [34], [33]. All this data was normalised in respect to the intracranial volume of each patient.

The following table illustrates the amount of features of each type and their missing rates

Table 4.1: Average Percentage of Missing Data for groups of features of each Biomarker after the aforementioned processing. (1) - Evenly Spaced Dataset, (2) - Unevenly Spaced Dataset. (*) - averaged from 28 features with 22.42% of missing data and 1 feature with 55.70% of missing data , (**) - averaged from 28 features with 16.07% of missing data and 1 feature with 50.69% of missing data

Feature	Post Processing Feats.	Missing Data (%)$^{(1)}$	Missing Data (%)$^{(2)}$
Age, TI	2	0	0
Diagnosis	1	0	0
MMSE	1	0.21	0.20
RAVLT	1	1.18	0.89
CDR	1	0.93	1.07
ADAS 11	1	0	0
ADAS 13	1	0.45	0.33
MRI	29	23.47$^{(*)}$	17.26$^{(**)}$

The evenly spaced dataset consisted of 429 patients due to how constraining it was to have five consultations for each patient, all temporally spaced 12 months from each other. This TI was chosen to prevent situations where data would be not be acquired due to acquisition protocols. As visible in Fig. (4.2), for example, non-AD patients had a protocol that did not allow for MRI images of the 18th month, thus disallowing a temporal spacing of six months. Moreover, after the second year, every protocol has acquisitions at least every 12 months, making this number a natural choice. The unevenly spaced

dataset does not suffer from the previously highlighted restrictions imposed from requiring constant TI and thus consisted of 944 patients.

The following tables illustrate the distribution of the diagnosis of the patients across consultations, for each dataset.

Table 4.2: Number of patients per diagnosis and consultation of the LSTM Dataset

Consultation	Diagnosis		
	CN	MCI	AD
1	101	326	2
2	99	318	12
3	106	273	50
4	102	250	77
5	105	223	101

Table 4.3: Number of patients per diagnosis and consultation of the T-LSTM Dataset

Consultation	Diagnosis		
	CN	MCI	AD
1	304	630	10
2	299	632	13
3	299	607	38
4	308	550	86
5	296	509	139

As visible above, the T-LSTM network had access to many more consultations than the standard LSTM. This was due to the formulation of this network, which allows the processing of longitudinal data with uneven temporal spacing. These tables also highlight that the number of patients suffering from AD increased as time passed, no matter if the time elapsed between consultations was constant or not. Lastly, in either dataset and any consultation, the MCI class was the most present one. The AD class was barely present at the first consultations in the resulting datasets, as it was not possible to find many even or unevenly spaced patients who started with and AD diagnosis at the first consultation and had data for a total of five subsequent consultations. On the other hand, the number of AD patients increased steadily throughout the consultations.

5

Classification Task and Results

Contents

In this chapter, the implementation details of each imputation method will be discussed. This will be followed by an evaluation of the quality and particularities of each imputation method. Lastly, the classification task, which is the main focus of this work, will be analysed, from the metrics evaluated, to the RNN models and parameters and lastly the results obtained in this task and their discussion.

5.1 Details of the Imputation Methods

In this section, the implementation details of each imputation method except for Masking, due to its simplicity, will be discussed and presented.

All data with numeric values (i.e. the non-missing data, hereby called X), which serves as input to the several methods, was normalised in respect to each feature following the expression below

$$X^{'} = \frac{X - X_{min}}{X_{max} - X_{min}}. \tag{5.1}$$

And thus $X^{'} \in [0, 1]$. The missing data, which each method handles differently, has the value of nan (not a number). Every method with exception of the GAIN had its data normalised this way.

5.1.1 Forward Imputation

The Forward Imputation is a simple method to fill in missing data by carrying forward the last known instance of data. In the implementation used in this work, in an instance where the missing data had no previously known value, the first (posterior) known value was carried backward. Patients were removed from the imputed dataset in case there existed no values of at least one feature across all of the five consultations.

5.1.2 Mean Imputation

An intuitive method where the missing values are replaced by the mean of all known values of that feature for a given patient. Notice that, similarly to the Forward Imputation, when there were no values for a feature across all consultations, the patient was removed from the imputed dataset. In a situation where only one value exists for any of the consultations, this method yields the same result as the Forward Imputation.

5.1.3 Random Imputation

Another method to fill in missing values is to introduce random values within reasonable bounds. The implementation used in this work replaced the missing values with random values between the minimum and maximum of each feature across all patients. This means that the interval of values that can be randomly chosen for imputation depends on the minimum and maximum across all patients for that given feature.

5.1.4 β-VAE Imputation

This method, which was only applied in the LSTM dataset, used a β-VAE from which samples of the learned latent space were used to perform the imputation of the missing data. The implementation, which had its sampling layer based on the Keras library website implementation [35] and the structure of its encoder and decoder based on an LSTM Autoencoder [36], had two layers of LSTM for the encoder (128 and 64 units, respectively) as well as for the decoder (64 and 128 units respectively). The possibility of using a value of β was added to the aforementioned implementation. This value was set to 0.1, as it was empirically leading to the best results (i.e. with the least MSE in the experiment in the Imputation Quality section).

5.1.5 GAIN Imputation

The GAIN learned the underlying distribution of the data and used this knowledge to perform the imputation of the missing data. The used GAIN was an adaptation of the code available online [37], with the parameters of hint rate set at 0.9 and α at 12.5 as they produced the best results in the experiment in the Imputation Quality section. Both the G and D were composed of an LSTM layer of 128 units followed by a fully connected layer.

All non-missing data X that served as input to this method was normalised with the following linear transformation

$$X^{'} = (b - a)\frac{X - X_{min}}{X_{max} - X_{min}} + a, \tag{5.2}$$

in which $b = 1$ and $a = 0.01$ and thus the normalised data fit in the interval of $X^{'} \in [0.01, 1]$. This interval was chosen due to the missing data taking the value of 0 (instead of nan) in this implementation. Notice that the value of a was arbitrarily chosen as a small enough non-zero value and any other similar value could have been chosen instead.

5.2 Imputation Quality

With the goal of comparing how good each imputation method is, an experiment to determine the imputation error of the VAE, of the GAIN, of the Mean Imputation and of the Forward Imputation with the increase of missing data was set up. Forming a set of 132 patients with 5 consultations, all with full data for the MRI and with each feature normalised between 0 and 1, it was possible to ”hide” the real values of each consultation (henceforth called introduced missing data) and to use the dataset with hidden values as input for each imputation method and then compare the output of the imputation techniques with the original value that was hidden.

For the sake of being able to draw comparisons between the error of every method, no patient had all their consultations hidden because, in such a situation, only the β-VAE and the GAIN would have been able to perform any imputation at all. This capacity of the β-VAE and the GAIN based imputations is inherent to their generative nature. Thus the maximum percentage of removable data was capped at 80% (4 out of 5 consultations for every single patient).

A missing rate of 10% would mean that 10% of all consultations had all their features hidden. The consultations which were hidden were chosen randomly and only following the previously mentioned ruled.

The β-VAE of composition as early described was trained for 1000 epochs, only on the known data after hiding values, and with the Root Mean Squared Error as loss function, as it yielded slightly better results than the standard MSE loss function.

The GAIN network parameters were as mentioned before, but with a modification where different learning rates were used such that the G has a learning rate of 0.01 and the D of 0.001. These parameters were more suited to ensure results with G and D convergence. Similarly to the β-VAE, this network too was trained with only the known data after introducing missing data.

The MSE of each imputation technique was calculated as follows

$$\text{MSE} = \frac{1}{n}\sum_{i=0}^{n}(X_i - \tilde{X}_i)^2 \tag{5.3}$$

Where X_i was a value (of any feature) which was hidden and $\tilde{X}_i$ was the value that was predicted by that imputation. Notice that n refers to the amount of values ”hidden” (thus a 10% missing rate has a lower n than a 20% missing rate and so on). To further clarify, this MSE was calculated only in respect

to the values which were hidden in each iteration of the experiment.

The following figure illustrates the results of this experiment with missing rates varying between 10 and 80%

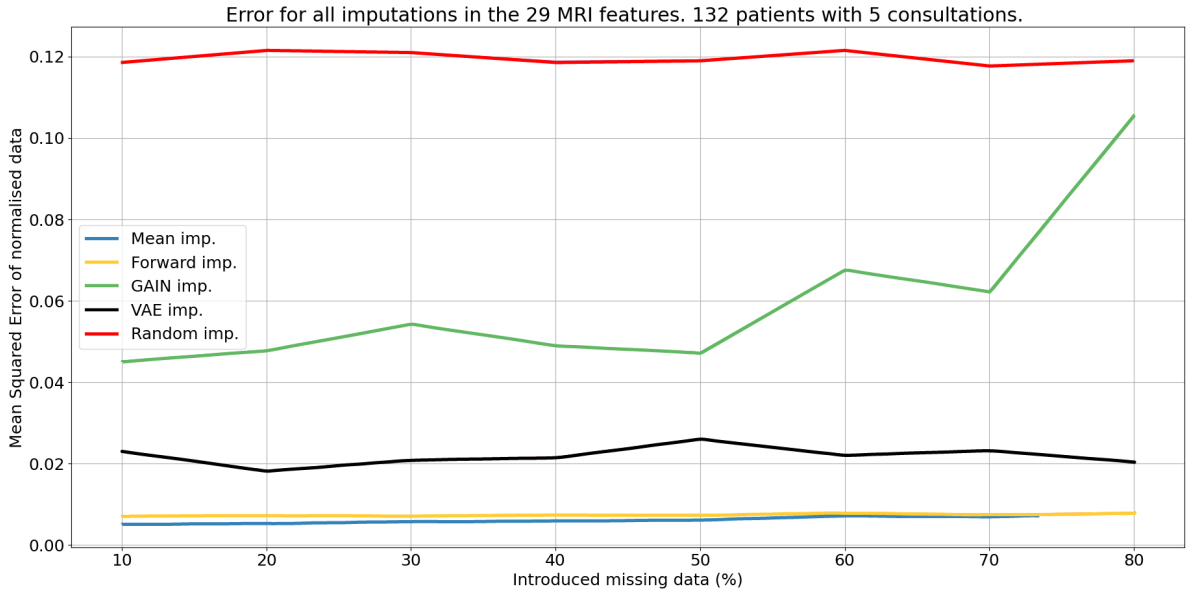

Figure 5.1: MSE of the Imputation Errors of the MRI features throughout different percentages of introduced missing data.

In addition to this last figure, another relevant metric was the standard deviation of each imputation

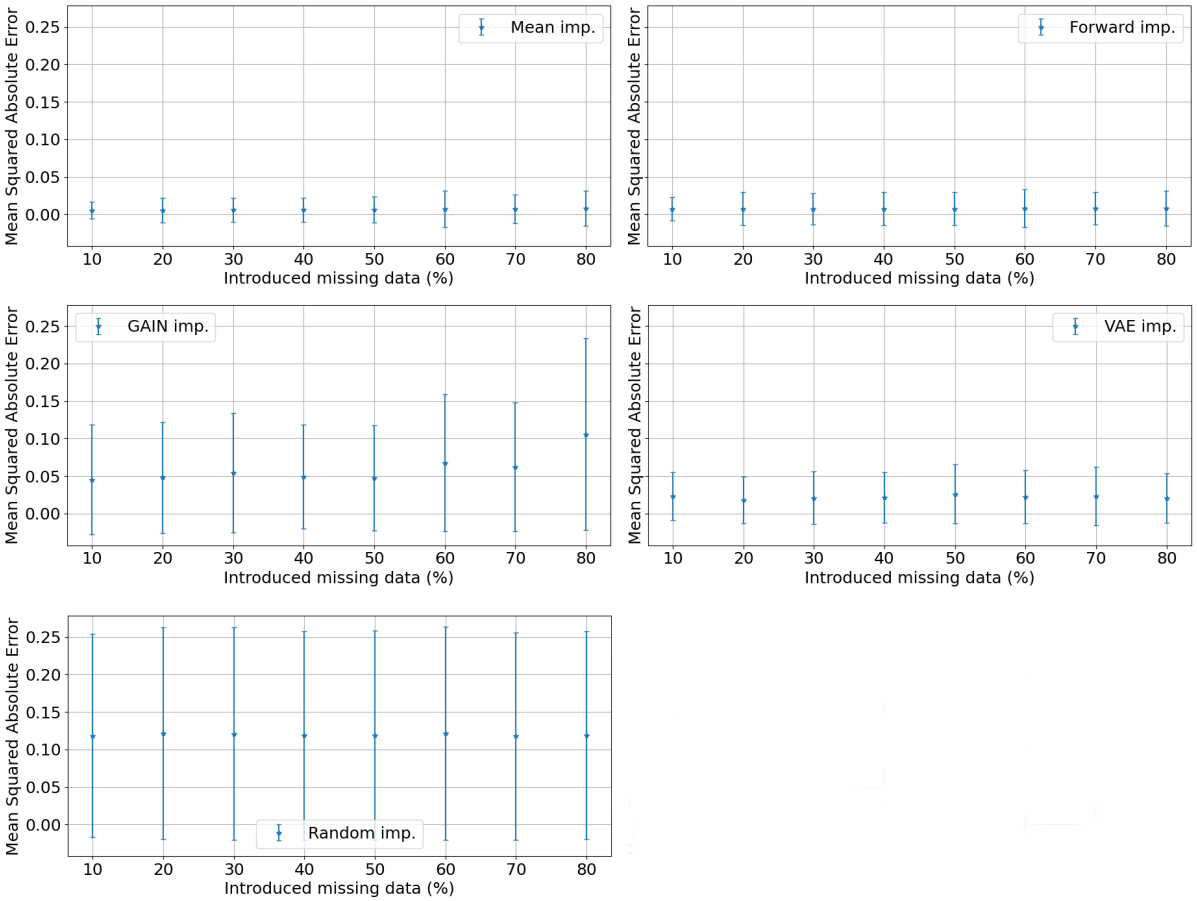

Figure 5.2: Standard Deviation of the Imputation Errors of the MRI features throughout different percentages of introduced missing data.

The Mean and Forward imputations had the best performance as they had the smallest MSE throughout all percentages of introduced missing data, as well as the least standard deviation. Their similar performances are explained by each feature not having a high rate of variation between consultations. These two methods output similar values and consequently have similar MSE and standard deviation. The β-VAE performed slightly worse than those two methods, as it always had a larger MSE and standard deviation. The GAIN performed the poorest of these four methods. It had a large and increasing MSE with the increase of the rate of missing data, as well as the highest standard deviation of these four methods for all rates.

The Random Imputation performed the worst out of all methods in both metrics, as expected. The standard deviation of this method was the same for any rate of missing data, as the values were randomly chosen in every single instance. This method provided an upper bound to the imputation error and standard deviation. The results of the GAIN and of the Random Imputation performed similarly for the

highest rate of missing data, leading to the hypothesis that the method does not produce valid results in this particular situation.

Another way to evaluate the results of this experiment was to compare the values of each feature after imputation by each method throughout the several rates of missing data. To represent these results, the following figures were made

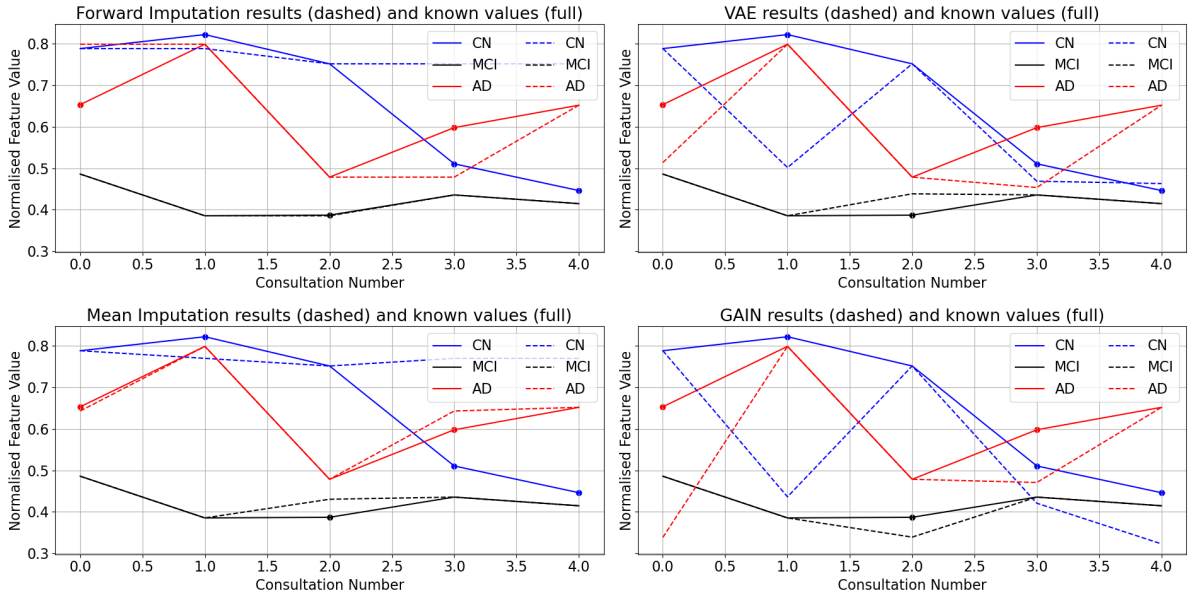

Figure 5.3: Imputation results for all 4 methods with 10% introduced missing rate.

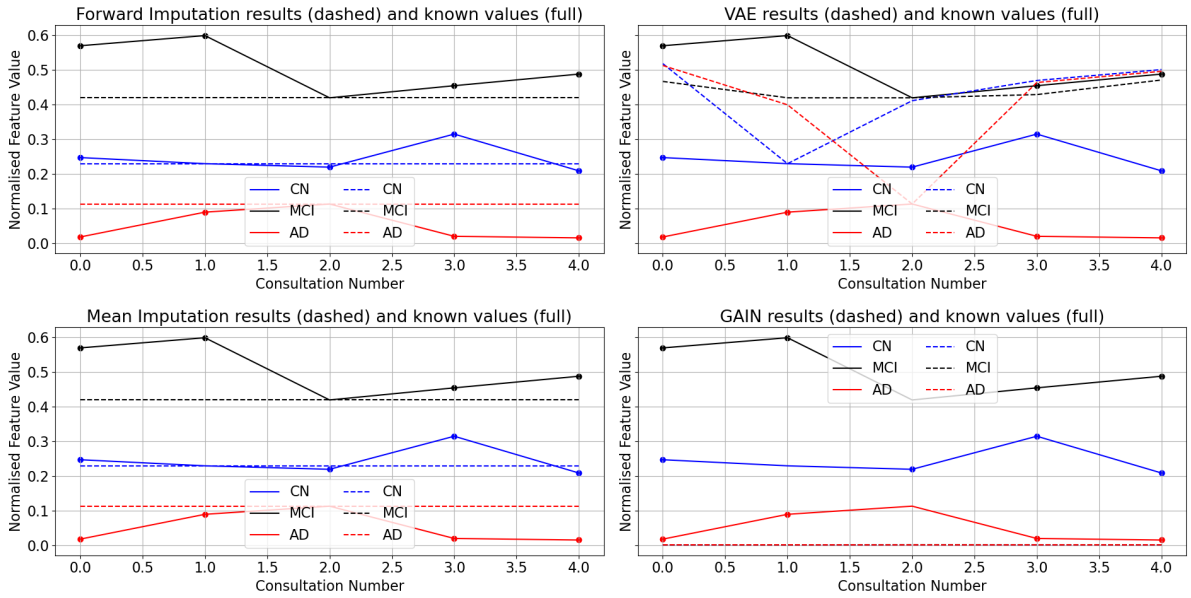

Figure 5.4: Imputation results for all 4 methods with 80% introduced missing rate.

Notice that both figures include the same feature throughout the 5 consultations. The feature was randomly chosen out of the 29 MRI features, and it is the Volume (WM Parcellation) of the Right Hippocampus. Also, the Random Imputation was excluded from this particular analysis as the quality of its results was random.

Each one of the plots within each figure corresponds to one imputation technique, appropriately identified in the titles. Every single plot also has data for the same feature of three patients with different diagnosis which underwent imputation. Blue stands for CN patients, black for MCI patients and red for AD patients. The full lines correspond to the known values while the dashed lines to the sequence output with each imputation technique. The full dots mark the consultations of each individual sequence that were hidden in order to perform imputation, and thus the instances where the data was not missing (and thus was not imputed), have no dot. Naturally, the further away the dashed line was from the full dot the greater the imputation error. The error was only calculated with the consultations marked with dots, as the imputations only produce results for those instances. The first figure had less dotted lines and less full dots as the missing rate of data was only 10%, which was the minimum introduced missing data for this experiment. Meanwhile, the second figure had the highest missing rate possible and thus has more full dots and more dashed lines.

In the first figure and with a missing rate of 10%, it is possible to see how potentially poorly the Forward Imputation and the Mean Imputation can perform, as in this particular case, in the CN patient, it had the worst performance out of the four. The errors of the generative models are also quite visible but, in general, they are not as pronounced as the errors of the Forward and Mean Imputations.

In the second figure it is highlighted the situation which was mentioned when describing the imputation methods. Both Forward and Mean Imputations produced the exact same results as there is only one consultation with results and in such circumstances, imputing with the mean will be the same as carrying it forward with the used implementation. The GAIN performs poorly to the point of having a mode collapse due to not having much data to train with. This is visible as the GAIN started outputting always the same value of approximately 0.1 for every single instance. On the other hand, the β-VAE had a non-constant output for every consultation and, despite the produced results being with relatively high deviations from the ideal values, they are outputs with a shape that is more similar to that of a real patient than the output of the Mean and Forward Imputations and thus, this method performed the best out of the four.

This analysis concludes that, while the Mean and Forward Imputations tend to perform best for any

rate of missing data, the β-VAE had a comparable performance throughout all rates of missing data. The GAIN, despite performing the worst out of them all, also has a small enough MSE to be tried out with the dataset. It also doesn't collapse until the missing rates across all features are around 50%, and thus can be used for this work, as the average missing rates of most features are, at most, around 20%. The generative methods have as biggest advantage the capacity to perform imputation when the regular methods cannot, such as in situations where all values of a feature are missing for a given patient. Thus these models are more appealing, as the larger the dataset after imputations, the better.

5.3 Classification Task

The classification task was performed for each one of the models and for each dataset after imputation. Each differently imputed dataset produced classification results with one label (CN, MCI or AD) per consultation. The results have three main metrics in which they were evaluated across the consultations:

- Accuracy.
- F1 Score for each class in a one vs all setting.
- Receiver Operating Characteristic (ROC) for each class in a one vs all setting and their respective Area Under Curve (AUC) where the True Negative Rate is plotted against the True Positive Rate.

The F1 Score was calculated for each model at each one of the k folds, while the ROC was calculated posterior to the concatenation of outputs of every model, meaning that this curve was singular for each imputed dataset. The F1 Score is calculated a total of k times and in the end the average F1 Score for each class was reported.

5.3.1 RNN models and their Parameters

Two models were designed in scope of this work with one main difference between each: one of the models used the LSTM as its RNN while the other one used the T-LSTM. Each model was composed of a Masking layer (which only had any impact when the masked datasets are the input), followed by two RNN layers of the same dimension that use the hyperbolic tangent as activation function, as it is often found to converge faster than other activation functions. They are then followed by a fully connected layer with softmax as its activation function, which produced the outputs, i.e., the diagnosis of each patient at every single consultation.

The first LSTM and the first T-LSTM layers of each model had 20% and 30% dropout at input, respectively. Both models have 10 % recurrent dropout on each layer, as these values yielded the best

results, and both models used as kernel and bias regularisers the L2 norm. All RNN layers had 256 units, as this amount of units was suggested to have yielded the best results through the state of the art [13], [14], [17], [19]. With these considerations, the LSTM based model had a total of 826115 parameters and the T-LSTM based model had 957699 parameters. The 15.93% more trainable parameters in the latter model are related to the differences that the T-LSTM model has which were discussed previously in Chapter (2).

The LSTM model was made using the Keras library while the T-LSTM model, with the code available online, was adapted for TensorFlow 2.0 code. The Adam optimizer [38] was used for both models but with different learning rates due to the Keras based implementation converging much quicker than the TensorFlow based one. Thus the LSTM model's learning rate was $1 \cdot 10^{-3}$ and the T-LSTM model's learning rate was $5 \cdot 10^{-6}$.

5.3.2 Training Process

Each dataset was divided into five folds which are then normalised, as four of the folds made the training set and the remaining fold made the test set in each of the five iterations. The training set was further split such that 10% of the total data in each iteration was used to form a validation set.

The model was trained using as cost function the categorical cross-entropy and the weights of the network that yielded the best results in the validation set were saved. This metric was being measured by obtaining the lowest validation loss during the training process.

Attending to the lack of AD patients throughout the consultations as seen in tables 4.2 and 4.3 causing some class imbalance, a known strategy of class weighting present in the scikit-learn library was used with the following formulation for each class

$$\text{class weight} = \frac{\text{samples}}{\text{number of classes} \cdot \text{instances of class}}. \tag{5.4}$$

This strategy assigned weights which were larger the less present a certain class was, and smaller the more present a class was.

5.3.3 Results for the Last Consultation

The obtained results were split between both models had emphasis on the fifth (and last) consultation. The main results will be presented with more detail on the last consultation but results will be shown for all consultations in the next section.

The LSTM model had less patients in its dataset but the β-VAE and GAIN imputed datasets available to use. Thus, the results obtained with the previously discussed metrics at the last consultation are reported as follows

Dataset (CN-MCI-AD)	**Test Accuracy (%)**	**Test F1 Score (% per class)**			**Test AUC (per class)**		
		CN	MCI	AD	CN	MCI	AD
Full (57-39-36)	80.26±4.64	**90.81±5.95**	59.42±9.99	79.17±4.20	0.961	0.847	0.956
Forward (86-65-64)	79.07±3.89	89.39±4.16	64.14±4.53	78.18±2.14	0.970	0.857	0.956
Mean (86-65-64)	76.28±7.11	87.86±5.36	56.85±11.68	77.18±6.21	**0.978**	0.865	0.904
Mask (105-223-101)	79.49±3.55	66.13±20.53	62.72±29.12	**83.42±2.54**	0.960	0.910	**0.972**
β-VAE (105-223-101)	81.11±3.47	73.13±19.01	**75.60±10.22**	79.95±3.85	0.967	0.913	0.969
GAIN (105-223-101)	**81.12±4.13**	70.66±19.92	72.55±17.29	82.23±3.08	0.969	**0.918**	0.971
Random (105-223-101)	77.16±2.80	69.68±18.80	64.01±22.39	79.14±2.02	0.960	0.894	0.963

Table 5.1: Average across all folds of the LSTM results of Accuracy and Test F1 Score and their respective Standard Deviations, as well as with the Test AUC which was calculated off the ROC.

As for the T-LSTM model, the results of the same metrics are as follows

Dataset (CN-MCI-AD)	**Test Accuracy (%)**	**Test F1 Score (% per class)**			**Test AUC (per class)**		
		CN	MCI	AD	CN	MCI	AD
Full (123-163-60)	74.28±1.48	89.08±1.76	**67.22±8.36**	60.68±3.94	0.968	0.864	0.905
Forward (180-229-96)	73.47±6.15	89.65±4.15	64.14±9.12	**63.47±6.18**	**0.978**	0.877	0.919
Mean (180-229-96)	72.67±5.62	**90.59±2.91**	63.17±2.60	61.15±6.90	**0.978**	0.865	0.904
Mask (296-509-139)	72.46±6.32	87.58±4.79	66.09±13.91	56.54±8.05	0.966	0.870	0.922
Random (296-509-139)	**73.94±5.90**	85.15±6.50	71.65±6.24	57.77±8.88	0.970	**0.882**	**0.924**

Table 5.2: Average across all folds of the T-LSTM results of Accuracy and Test F1 Score and their respective Standard Deviations, as well as with the Test AUC which was calculated off the ROC.

Notice that in parenthesis are the total number of patients with each diagnostic at the last consultation

in each imputed dataset. To exemplify, 57, 39 and 36 are the number of patients with CN, MCI and AD diagnosis, respectively, for the dataset containing only full data in Table 5.1. The best results for either model are highlighted in bold.

As discussed in Chapter 4, the LSTM model's dataset was smaller than the T-LSTM model's dataset and, despite this, the numbers of patients with each labels varied within each table. This is due to how each imputation method works. The Full dataset had the least patients, as it excluded every patient with any instance of missing data. The Mean and Forward imputed datasets had the same amount of patients, which was lower than the amount of patients present in the β-VAE and GAIN imputed datasets and in the Masked dataset. This occurred due to situations where all the values for at least one feature are missing and thus neither of the two former methods can produce a value to fill in the missing value.

Each table has a relatively similar range of results throughout methods and models. In the first table, the LSTM model's results can be placed in two groups: the first one with the Full dataset, Forward and Mean imputed datasets and the second one with the Masked, β-VAE, GAIN and Random imputed datasets. The former group had a generally lower standard deviation for the Test F1 Score while having lower values of both Test F1 Score and Test AUC for the MCI and AD classes. The latter group had higher standard deviation in the Test F1 Score in the CN and MCI classes and lower standard deviation in the AD class. Its test AUC values for the MCI and AD classes were also higher than for the first group.

In terms of Test Accuracy, the second group performs best due to having lower standard deviation and higher average values than the first group, which is especially true for both the β-VAE and GAIN imputed datasets. The condition that RNN have of requiring large datasets to output better results might also explain the performance difference of the two groups, as the second group of results performed best because the imputations at stake allowed for much bigger sets of data. The separation of the results in two was also a consequence of the size of the datasets of each group. Bigger datasets have generally much better performance for the MCI and AD classes. Across most metrics, the VAE Imputation yielded the best results, closely followed or even surpassed by the GAIN Imputation. The results from the β-VAE Imputation were considered best as they had lower standard deviation than the GAIN Imputation and the highest overall F1 Score values.

Attending to the results of the Imputation Quality section, namely to Figs. 5.1 and 5.2, it would be expected that the Mean and Forward Imputations would perform similar to one another and outperform the β-VAE Imputation, which, in its turn, would outperform the GAIN Imputation. This is based on previously obtained results for the MSE and standard deviation of each method. In practice however, the

generative methods allowed for bigger datasets and thus outperformed the simple methods. The total of patients for either the β-VAE or GAIN imputed datasets was in fact twice as big than the Mean or Forward imputed datasets and with proportionally more samples of MCI patients. This dataset size difference explains the performance differences of the aforementioned methods. Another factor is due to how the models work. Despite having more MSE and standard deviation as in the Imputation Quality section, the results that the β-VAE and GAIN produced follow a pattern which is similar to the one of a human patient rather than just the constant values that the Forward and Mean imputations produced, as highlighted in the situation of Figure 5.4.

The second table contains the T-LSTM model's results. The Mean imputed dataset had the lowest standard deviation across the metrics while the Masked dataset had the highest standard deviations for almost every metric. Across all metrics, the Forward Imputation yielded the best results and was closely followed by the Random Imputation, especially in terms of F1 Score. This was due to how data in the dataset starts missing in later consultations rather than in earlier ones and thus, with the progression of AD, which is irreversible, an imputation technique that uses the last known value is more prone to yield better results than one that averages all known consultations for the same patient, like the Mean imputation. This reasoning on the differences in performance between the Forward and Mean imputed datasets applies for both models, as in either dataset they also had higher missing rates in later consultations. It is also worth to emphasise that the number of patients in this model was much higher than in the LSTM model. This was due to the possibility of using non-equally temporally spaced patients that this model has.
In both tables, the Masked datasets had the highest overall standard deviation rates in the Test F1 Score, despite not performing poorly in any other metric. This is explained by the amount of data that this method masks. The more instances of data that were masked, the less data the network used in its training process. As RNN perform better the more data they have available to be trained on, having less data available worsens their performance and thus the performance of Masking.
Comparing the two tables, the Test Accuracy was lower in the T-LSTM model and this model had a generally higher standard deviation. The Test F1 score was more often higher with a lower standard deviation for the CN and MCI classes for the T-LSTM model. On the other hand it was overall lower with a higher standard deviation for the AD class. The Test AUC was similar for the CN class and slightly lower on both the MCI and AD classes for the T-LSTM model. This means that, even though the T-LSTM model had more patients in its datasets and used the TI between consultations, it did not yield better results for the AD class, despite performing better for the other two classes. This effect happens as almost no patient began with an AD diagnosis and, as the TI value was only zero at the first consultation, all the subsequent consultations had their weights discounted due to how the Adjusted Previous Memory was

calculated, as seen in Equation 2.11. This way, the learned generalisation for this class was of lower quality than in the LSTM model.

While the AUC were presented, the ROC from which they were calculated are plotted separately in the following figures for the LSTM and T-LSTM models, respectively

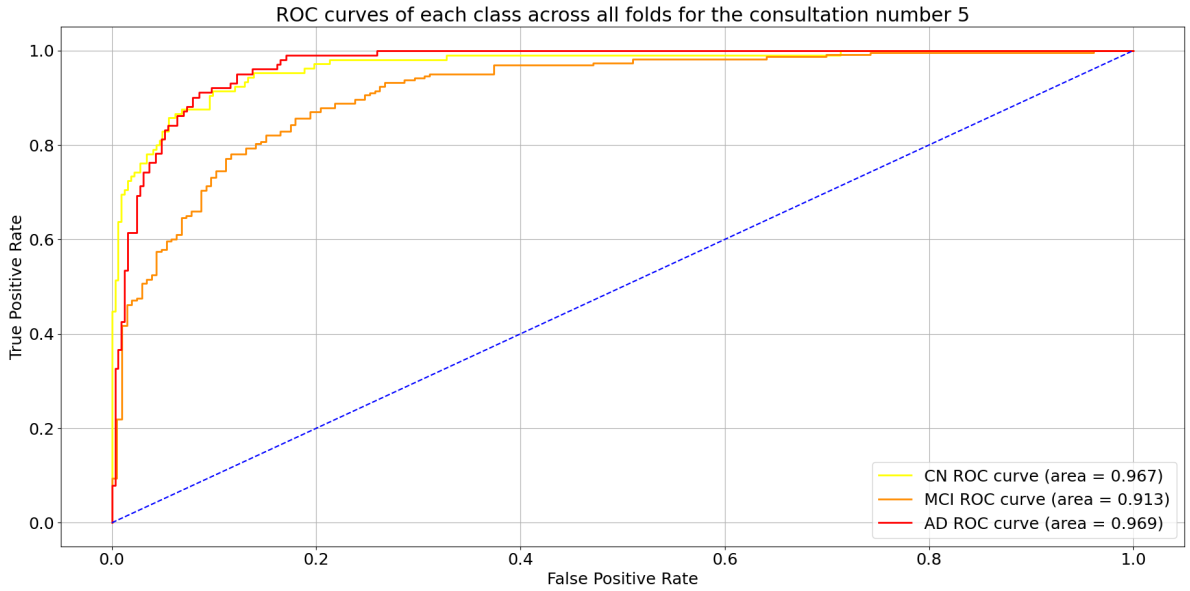

Figure 5.5: ROC of the three classes at the 5th consultation for the VAE imputed dataset,

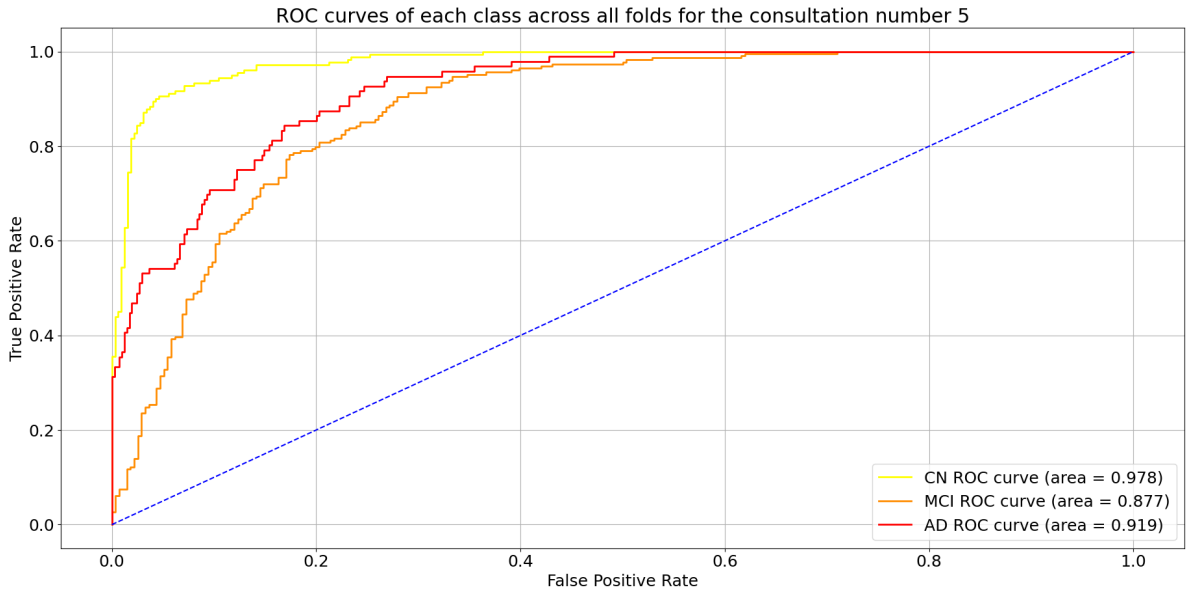

Figure 5.6: ROC of the three classes at the 5th consultation for the Forward imputed dataset.

The CN curve had the most AUC, followed by the AD curve. The MCI curve performed always worse

than the other two but much better than a random classifier would (marked in the figures with the blue dashed line). These particular figures were selected as they corresponded to the best results in the tables, which are marked in bold in Tables 5.1 and 5.2, respectively. However, the remainder of the ROC plots are very similar and will be only shown in Appendix A.

Another way to present these results which has a better graphical visualisation is through the usage of boxplots, side by side, of each kind of metric for each dataset, as follows

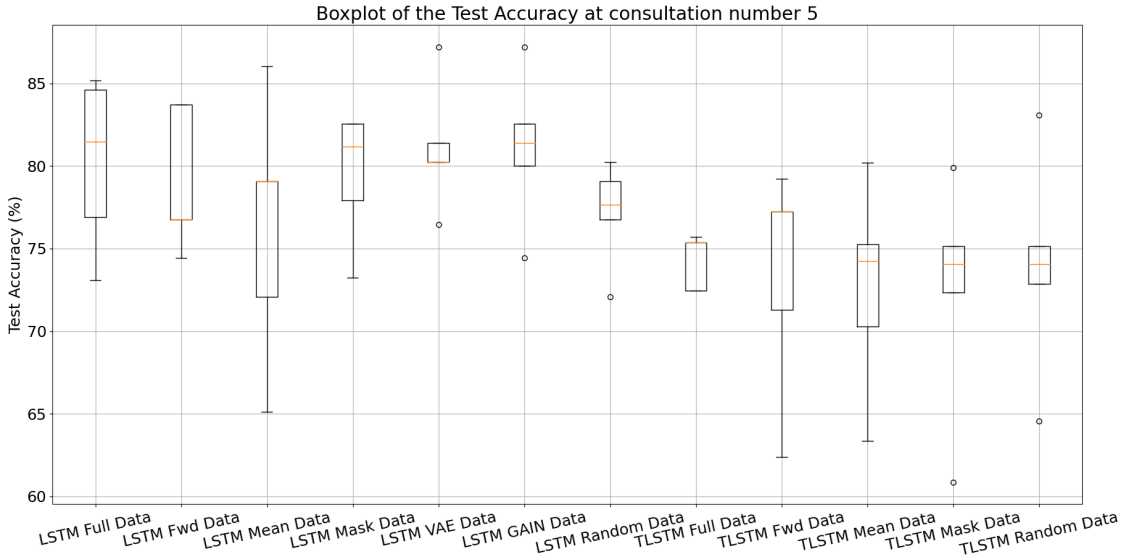

Figure 5.7: Boxplots of the Test Accuracy each method side by side. The yellow lines mark each median.

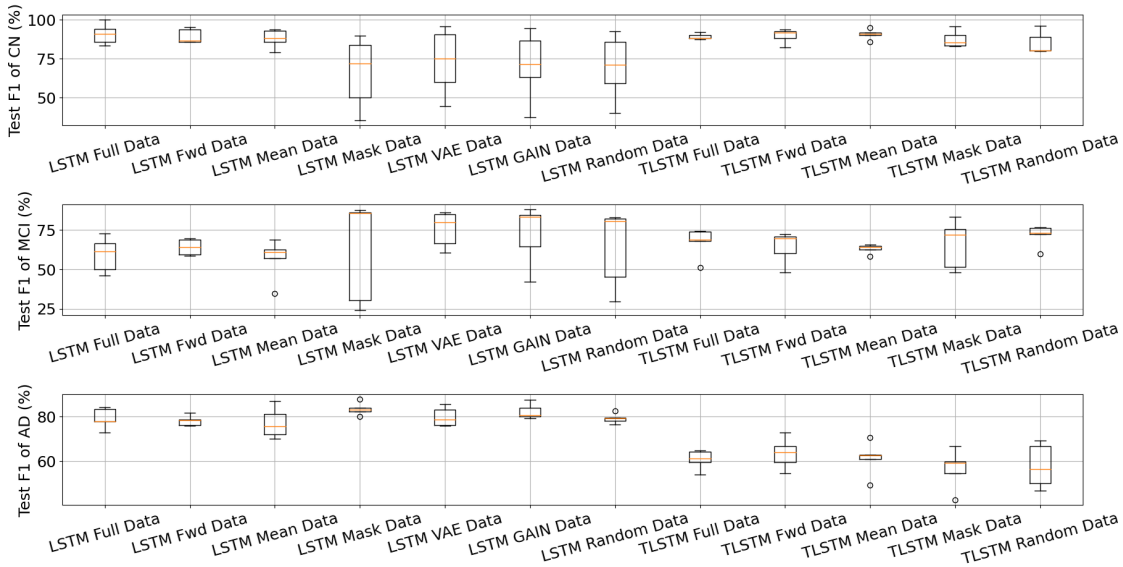

Figure 5.8: Boxplots of the F1 Score of each class and each method side by side. The yellow lines mark each median.

The first boxplot highlights that the Test Accuracy, in median, was higher for the LSTM model and had less standard deviation. This representation also gives a visual intuition to the negative impact that higher standard deviations and outliers have to the Test Accuracy.

The second boxplot reinforces what previously was discussed, namely that, for the CN and MCI classes, the T-LSTM model produced better results while for the AD class the LSTM model was best.

Finally, in either model, through the previous tables and figures, it is concluded that the Random Imputation is always competitive with both the baseline methods and the proposed methods. While this doesn't invalidate the comparisons between the different imputations, it leads to a conclusion that imputation was not necessary for the datasets used in this work.

5.3.4 Results for all Consultations

The models presented before had many parameters as a consequence of having models that could predict the diagnosis at each consultation. While the emphasis is on the last consultation, the results for the remaining consultations were obtained too. The results on the following tables contain the results of each metric from consultations one to five of the VAE imputed dataset for the LSTM model and of the Forward imputed dataset for the T-LSTM model

Consultation (CN-MCI-AD)	**Test Accuracy (%)**	**Test F1 (% per class)**			**AUC (per class)**		
		CN	MCI	AD	CN	MCI	AD
1 (101-326-2)	98.60±0.87	94.06±7.68	98.88±0.88	- (1)	0.998	0.979	0.335
2 (99-318-12)	93.00±3.91	81.06±24.16	94.52±3.81	33.33±0.00	0.989	0.957	0.828
3 (106-273-50)	81.57±5.12	82.52±21.17	80.08±11.73	59.44±6.27	0.979	0.920	0.938
4 (102-250-77)	80.64±3.18	66.69±29.89	75.33±13.51	72.18±5.99	0.979	0.916	0.959
5 (105-223-101)	81.11±3.47	73.12±19.01	75.60±10.22	79.95±3.85	0.967	0.913	0.969

Table 5.3: Average across all folds for each consultation of the LSTM model for the VAE imputed dataset. Results of Accuracy, F1 Score and AUC calculated off the ROC. In parenthesis are the number of patients with each diagnostic at the last consultation. (1) - not calculated due to not enough patients

Consultation (CN-MCI-AD)	**Test Accuracy (%)**	**Test F1 Score (% per class)**			**AUC (per class)**		
		CN	MCI	AD	CN	MCI	AD
1 (188-307-10)	87.72±3.41	87.72±5.20	89.37±2.30	30.86±24.69	0.961	0.941	0.948
2 (187-305-13)	90.69±1.84	92.78±1.36	91.95±1.76	36.33±14.93	0.987	0.965	0.911
3 (189-290-26)	84.75±4.50	95.00±3.59	85.90±4.00	29.81±12.72	0.992	0.948	0.869
4 (190-251-64)	75.84±7.37	93.57±3.19	71.53±5.76	52.72±12.43	0.988	0.908	0.888
5 (180-229-96)	73.47±6.15	89.65±4.15	64.14±9.12	63.47±6.18	0.978	0.877	0.919

Table 5.4: Average across all folds for each consultation of the T-LSTM model for the Forward imputed dataset. Results of Accuracy, F1 Score and AUC calculated off the ROC. In parenthesis are the number of patients with each diagnostic at the last consultation.

Notice that the number of patients of each class is identified the same way as in Tables 5.1 and 5.2.

The two tables show that, for either model, the Test Accuracy decreased as the consultation number increased. The same happened with the Test F1 Score of the CN and MCI classes. By contrast, the Test F1 Score of the AD class increased with the number of consultations. Generally speaking, the AUC did not change much between consultations. Intuitively, one would expect that the metrics would all

improve with the increase of consultations but in practice the opposite has happened, that is, the first consultations had generally better metrics than the final ones.

Attending to the number of patients, it is visible that in either model, but more particularly so in the LSTM model, that the number of AD patients was much lower until the third consultation. In fact, the first consultation of the LSTM model was almost a binary classification problem where the patient was in either the classes of CN or MCI. As consultations go on, and with more AD diagnosed patients in this dataset, the model started to learn to classify patients with AD. The T-LSTM model suffered from the same problem, but instead the two first consultations had almost the same number of AD patients and thus the Test Accuracy increased from consultation 1 to 2, but then started dropping as the number of AD patients started steadily increasing.

The following figures show the Test Accuracy throughout each consultation for each model and each dataset

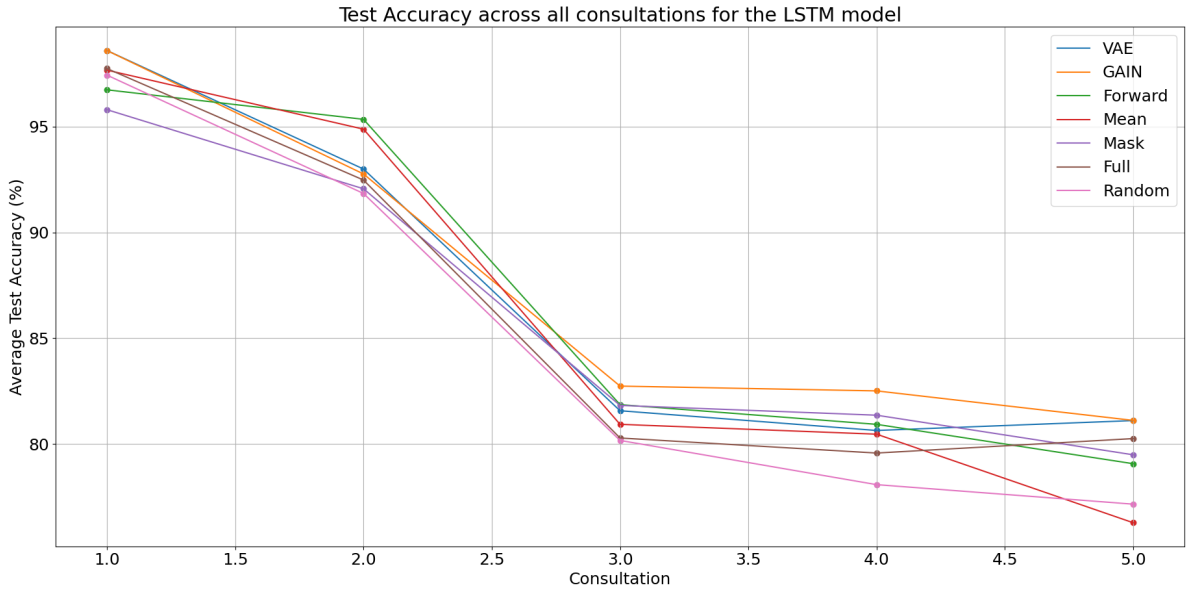

Figure 5.9: Test Accuracy of each method throughout consultations for the LSTM model.

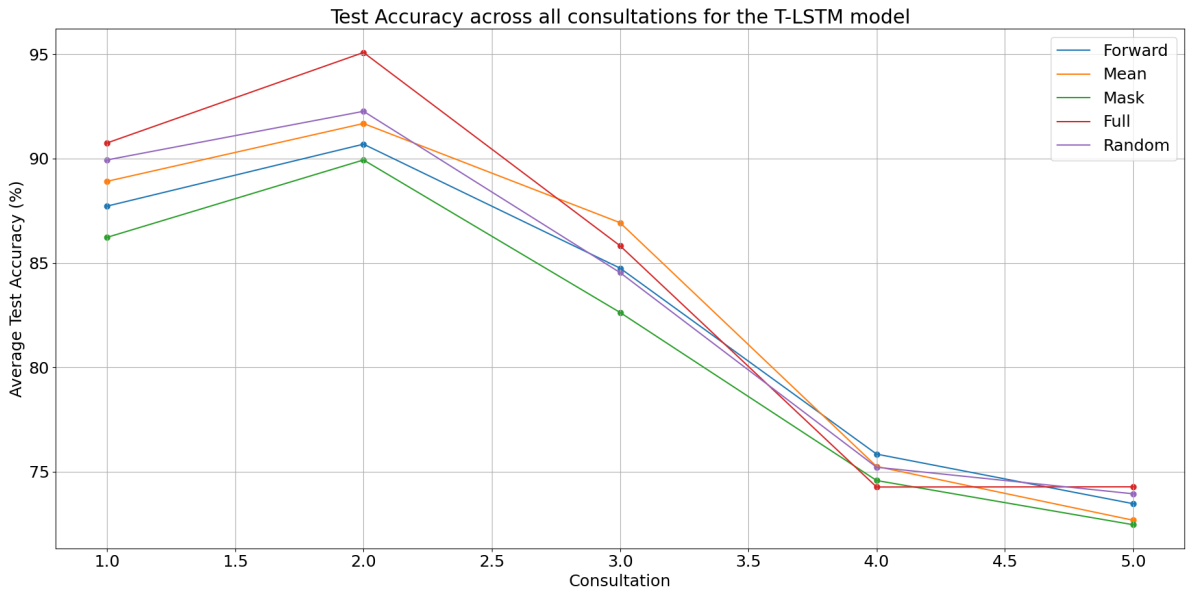

Figure 5.10: Test Accuracy of each method throughout consultations for the T-LSTM model.

As visible above, for either model the Test Accuracy was highest in the first two consultations. In either model the imputed datasets used all have roughly the same distribution of patients, i.e., in the LSTM model's dataset the first consultation barely had any AD patients and the T-LSTM model's dataset had

roughly the same AD patients in the two first consultations. This explains why the behaviour of the datasets followed the same trend within each model. These figures also highlight that, Test Accuracy wise, the LSTM model performed better than the T-LSTM throughout the progression of the consultations. At the third consultation, the LSTM slowed down the rate at which its Test Accuracy reduced, while the T-LSTM kept steadily losing Test Accuracy after the second consultation.

Lastly, Appendix B contains the ROC plots of every consultation for the LSTM model with the β-VAE imputed dataset and for the T-LSTM model with the Forward imputed dataset.

6

Conclusions and Further Work

Contents

6.1 Conclusions

In the present setup, the generative methods produced results which were competitive, in terms of imputation quality, with the standard widely used methods, especially for lower rates of missing data. These results could have been more competitive with the state of the art but they show potential, especially the β-VAE and GAIN.

In the course of this work, the obtained results were comparable to the state of the art, surpassing some results in terms of Test Accuracy while employing less patients than in the work of M. M. Ghazi et al., 2019 [19]. On the other hand, not using all relevant features for this disease hinders the results of this work and brings its Test Accuracy down. The CSF and PET measurements, which had too high missing rates, as previously discussed, could not be used. In part, this lack of features was a reason why the obtained results were behind some other state of the art results such those of Z .C Lipton et al. 2016 [14]. It was also the main reason why the Random Imputation performs competitively with the other imputations.

One of the advantages of a T-LSTM based model was that, through the usage of the TI, it enabled more patients to be used in its dataset. Despite the much larger amount of patients, the results were not better across all metrics than those of an LSTM based model. In fact, this novel model had lower Test Accuracy than the LSTM model. Despite having had better F1 Scores for the CN and MCI patients, the T-LSTM model performed worse for AD patients in this metric. Some of the performance differences between the results of the two models are explained with the dataset size differences, namely the differences in standard deviation between the two models. However, the mechanism of the Adjusted Previous Memory, with its current formulation, could potentially be reducing the Test Accuracy of this model and the F1 Score of the AD class. This said, considering how RNN need large datasets, models of networks such as the T-LSTM allow for usages of more data than the LSTM, which is a big advantage.

As the remaining consultations are introduced, a pattern emerged. The results were relatively good in the first two consultations due to the problem being between a binary classification task, as there are almost no AD patients in the datasets of either model so early. At the third consultation onward, the classification task became more difficult as the MCI patients developed AD and, from that point on, there are more AD patients. Datasets where all three classes would be present throughout every stage without major class imbalances would make mitigate this situation.

6.2 Further Work

There are, however, some changes which could be made to this work and potentially could be relevant not just for these methods but also for handling longitudinal data with RNN in general.

The same way that each class can be attributed weights with formulations such as the one in Equation 5.4, the same school of thought could be applied to weight each sequence (or, in this case, for each consultation). At the moment there were not any examples of this concept in the literature and thus different ways to weight each sequence could be theorised, for example, in the following ways:

- Each sequence's weight could depend on relevant features of the data such as, to list some examples, the age of a patient, the TI between consultations, the label(s) of that sequence, or some (non) linear decay function.
- Each individual sequence's weight depending on characteristics typical of the task. To exemplify, in AD related works, it would make sense that some consultations are more key to diagnose someone with this pathology, and thus these said key consultations would have larger weights than the others.
- The sequences of patients that, for example, develop AD despite prior MCI or CN diagnoses, could have a larger weight than the sequences of patients that have a constant diagnosis for every consultation. That is, sequences where there is an evolution of a patient's status could be given more importance in the model's learning.

Bibliography

[1] F. Chollet, *Deep Learning with Python*. Manning Publications Co., 2018.

[2] X. Yuan, L. Li, and Y. Wang, "Nonlinear Dynamic Soft Sensor Modeling With Supervised Long Short-Term Memory Network," *IEEE Transactions on Industrial Informatics*, vol. PP, pp. 1–1, 02 2019. [Online]. Available: http://dx.doi.org/10.1109/TII.2019.2902129

[3] I. M. Baytas, C. Xiao, X. Zhang, F. Wang, A. K. Jain, and J. Zhou, "Patient Subtyping via Time-Aware LSTM Networks," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 65–74. [Online]. Available: https://doi.org/10.1145/3097983.3097997

[4] Y. Luo, X. Cai, Y. ZHANG, J. Xu, and Y. Xiaojie, "Multivariate Time Series Imputation with Generative Adversarial Networks," in *Advances in Neural Information Processing Systems*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds., vol. 31. Curran Associates, Inc., 2018. [Online]. Available: https://proceedings.neurips.cc/paper/2018/file/96b9bff013acedfb1d140579e2fbeb63-Paper.pdf

[5] J. Yoon, J. Jordon, and M. van der Schaar, "GAIN: Missing Data Imputation using Generative Adversarial Nets," in *Proceedings of the 35th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, J. Dy and A. Krause, Eds., vol. 80. PMLR, 10–15 Jul 2018, pp. 5689–5698. [Online]. Available: http://proceedings.mlr.press/v80/yoon18a.html

[6] J. Rocca and B. Rocca. Understanding Variational Autoencoders (VAEs) - Towards Data Science. (accessed: 05.02.2021). [Online]. Available: https://towardsdatascience.com/understanding-variational-autoencoders-vaes-f70510919f73

[7] Alzheimer's Disease Neuroimaging Initiative. (accessed: 13.04.2020). [Online]. Available: http://adni.loni.usc.edu/about/

[8] Alzheimer's Disease and Alzheimer's Dementia. (accessed: 13.04.2020). [Online]. Available: https://www.alzheimer-europe.org/Dementia/Alzheimer-s-disease-and-Alzheimer-s-dementia

[9] European Federation of Pharmaceutical Industries and Associations. (accessed: 13.04.2020). [Online]. Available: https://www.efpia.eu/we-wont-rest/innovation/alzheimer-s-disease/

[10] National Institute on Aging - Alzheimer's Disease Fact Sheet. (accessed: 29.05.2020). [Online]. Available: https://www.nia.nih.gov/health/alzheimers-disease-fact-sheet

[11] Alzheimer's Disease Neuroimaging Initiative - Data Types. (accessed: 25.01.2021). [Online]. Available: http://adni.loni.usc.edu/data-samples/data-types/

[12] A. R. T. Donders, G. J. [van der Heijden], T. Stijnen, and K. G. Moons, "Review: A gentle introduction to imputation of missing values," *Journal of Clinical Epidemiology*, vol. 59, no. 10, pp. 1087 – 1091, 2006. [Online]. Available: https://doi.org/10.1016/j.jclinepi.2006.01.014

[13] M. Nguyen, N. Sun, D. C. Alexander, J. Feng, and B. T. T. Yeo, "Modeling Alzheimer's Disease Progression using Deep Recurrent Neural Networks," in *2018 International Workshop on Pattern Recognition in Neuroimaging (PRNI)*, 2018, pp. 1–4. [Online]. Available: https://doi.org/10.1016/j.neuroimage.2020.117203

[14] Z. C. Lipton, D. Kale, and R. Wetzel, "Directly Modeling Missing Data in Sequences with RNNs: Improved Classification of Clinical Time Series," in *Proceedings of the 1st Machine Learning for Healthcare Conference*, ser. Proceedings of Machine Learning Research, F. Doshi-Velez, J. Fackler, D. Kale, B. Wallace, and J. Wiens, Eds., vol. 56. Northeastern University, Boston, MA, USA: PMLR, 18–19 Aug 2016, pp. 253–270. [Online]. Available: http://proceedings.mlr.press/v56/Lipton16.html

[15] S. Daberdaku, E. Tavazzi, and B. D. Camillo, "Interpolation and K-Nearest Neighbours Combined Imputation for Longitudinal ICU Laboratory Data," in *2019 IEEE International Conference on Healthcare Informatics (ICHI)*, 2019, pp. 1–3. [Online]. Available: https://doi.org/10.1109/ICHI.2019.8904624

[16] M. D. Samad and L. Yin, "Non-linear regression models for imputing longitudinal missing data," in *2019 IEEE International Conference on Healthcare Informatics (ICHI)*, 2019, pp. 1–3. [Online]. Available: https://doi.org/10.1109/ICHI.2019.8904528

[17] T. Wang, R. G. Qiu, and M. Yu, "Predictive Modeling of the Progression of Alzheimer's Disease with Recurrent Neural Networks," *Scientific Reports*, vol. 8, no. 1, p. 9161, 2018. [Online]. Available: https://doi.org/10.1038/s41598-018-27337-w

[18] Z. Che, S. Purushotham, K. Cho, D. Sontag, and Y. Liu, "Recurrent Neural Networks for Multivariate Time Series with Missing Values," *Scientific Reports*, vol. 8, no. 1, p. 6085, 2018. [Online]. Available: https://doi.org/10.1038/s41598-018-24271-9

[19] M. Mehdipour-Ghazi, M. Nielsen, A. Pai, M. J. Cardoso, M. Modat, S. Ourselin, and L. Sørensen, "Training Recurrent Neural Networks robust to incomplete data: Application to Alzheimer's Disease progression modeling," *Medical Image Analysis*, vol. 53, p. 39–46, 2019. [Online]. Available: https://doi.org/10.1016/j.media.2019.01.004

[20] H.-G. Kim, G.-J. Jang, H.-J. Choi, M. Kim, Y.-W. Kim, and J. Choi, "Recurrent Neural Networks with Missing Information Imputation for Medical Examination Data Prediction," in *2017 IEEE International Conference on Big Data and Smart Computing (BigComp)*, 2017, pp. 317–323. [Online]. Available: https://doi.org/10.1109/BIGCOMP.2017.7881685

[21] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative Adversarial Nets," in *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, ser. NIPS'14. Cambridge, MA, USA: MIT Press, 2014, p. 2672–2680. [Online]. Available: https://dl.acm.org/doi/10.5555/2969033.2969125

[22] D. P. Kingma and M. Welling, "Auto-Encoding Variational Bayes," in *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014. [Online]. Available: https://arxiv.org/pdf/1312.6114.pdf

[23] C. Doersch, "Tutorial on Variational Autoencoders," 2016. [Online]. Available: https://arxiv.org/pdf/1606.05908.pdf

[24] Y. L. Qiu, H. Zheng, and O. Gevaert, "Genomic Data Imputation with Variational Auto-Encoders," *GigaScience*, vol. 9, no. 8, 08 2020, giaa082. [Online]. Available: https://doi.org/10.1093/gigascience/giaa082

[25] V. Fortuin, D. Baranchuk, G. Raetsch, and S. Mandt, "GP-VAE: Deep Probabilistic Time Series Imputation," in *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, S. Chiappa and R. Calandra, Eds., vol. 108. PMLR, 26–28 Aug 2020, pp. 1651–1661. [Online]. Available: http://proceedings.mlr.press/v108/fortuin20a.html

[26] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Comput.*, vol. 9, no. 8, p. 1735–1780, Nov. 1997. [Online]. Available: https://doi.org/10.1162/neco.1997.9.8.1735

[27] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling," in *NIPS 2014 Workshop on Deep Learning, December 2014*, 2014. [Online]. Available: https://arxiv.org/pdf/1412.3555.pdf

[28] D. Cohen, K. Carpenter, J. Jarrell, and X. Huang, “Deep Learning-based classification of multi-categorical Alzheimer’s Disease data,” *Current Neurobiology*, vol. 10, pp. 141–147, 08 2019. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6889824/

[29] J. Neelaveni and M. S. G. Devasana, “Alzheimer Disease Prediction using Machine Learning Algorithms,” in *2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS)*, 2020, pp. 101–104. [Online]. Available: https://doi.org/10.1109/ICACCS48705.2020.9074248

[30] C. P. Burgess, I. Higgins, A. Pal, L. Matthey, N. Watters, G. Desjardins, and A. Lerchner, “Understanding disentangling in β-VAE,” *CoRR*, vol. abs/1804.03599, 2018. [Online]. Available: http://arxiv.org/pdf/1804.03599.pdf

[31] I. Shafkat. Intuitively Understanding Variational Autoencoders - Towards Data Science. (accessed: 02.01.2021). [Online]. Available: https://towardsdatascience.com/intuitively-understanding-variational-autoencoders-1bfe67eb5daf

[32] S. Kullback, *Information Theory and Statistics*. John Wiley & Sons Inc., 1959.

[33] TADPOLE - Data. (accessed: 29.05.2020). [Online]. Available: https://tadpole.grand-challenge.org/Data/#List

[34] Dementia symptoms and areas of the brain. (accessed: 29.05.2020). [Online]. Available: https://www.alzheimers.org.uk/about-dementia/symptoms-and-diagnosis/how-dementia-progresses/symptoms-brain

[35] F. Chollet. Variational Autoencoder. (accessed: 17.02.2021). [Online]. Available: https://keras.io/examples/generative/vae/

[36] C. Ranjan. Codebase for ”Understanding an LSTM Autoencoder”. (accessed: 01.04.2021). [Online]. Available: https://github.com/cran2367/understanding-lstm-autoencoder

[37] J. Yoon, J. Jordon, and M. van der Schaar. Codebase for ”Generative Adversarial Imputation Networks (GAIN)”. (accessed: 17.02.2021). [Online]. Available: https://github.com/jsyoon0823/GAIN

[38] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2015. [Online]. Available: http://arxiv.org/pdf/1412.6980.pdf

A

ROC for the 5th Consultation

ROC plots for the LSTM

Figure A.1: ROC of the 3 classes at the 5th consultation for the full dataset.

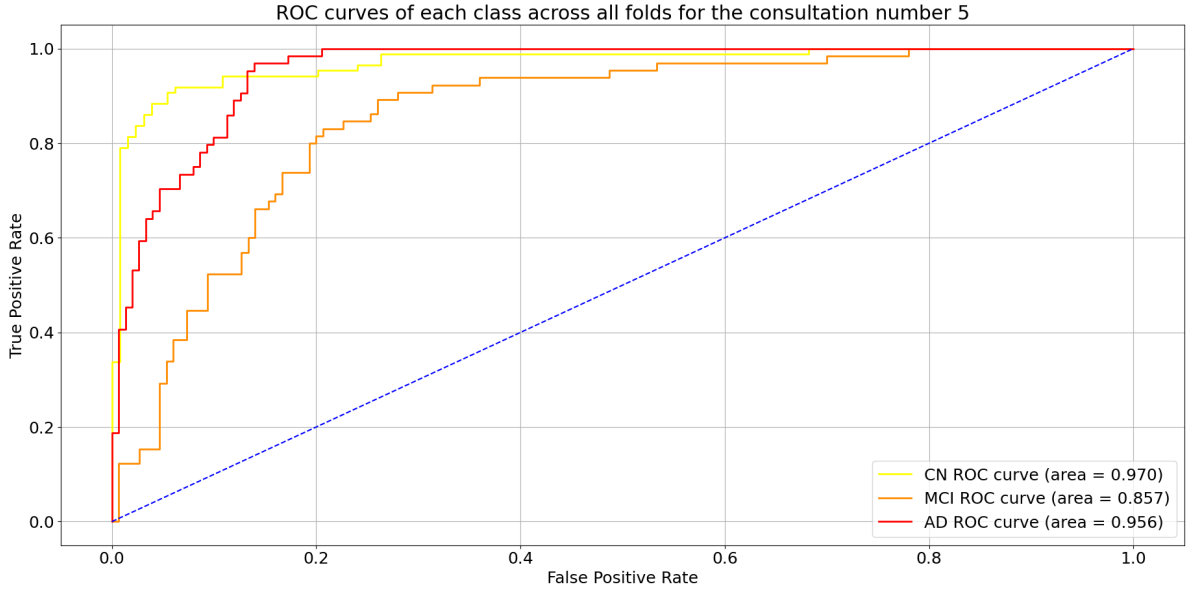

Figure A.2: ROC of the 3 classes at the 5th consultation for the forward imputed dataset.

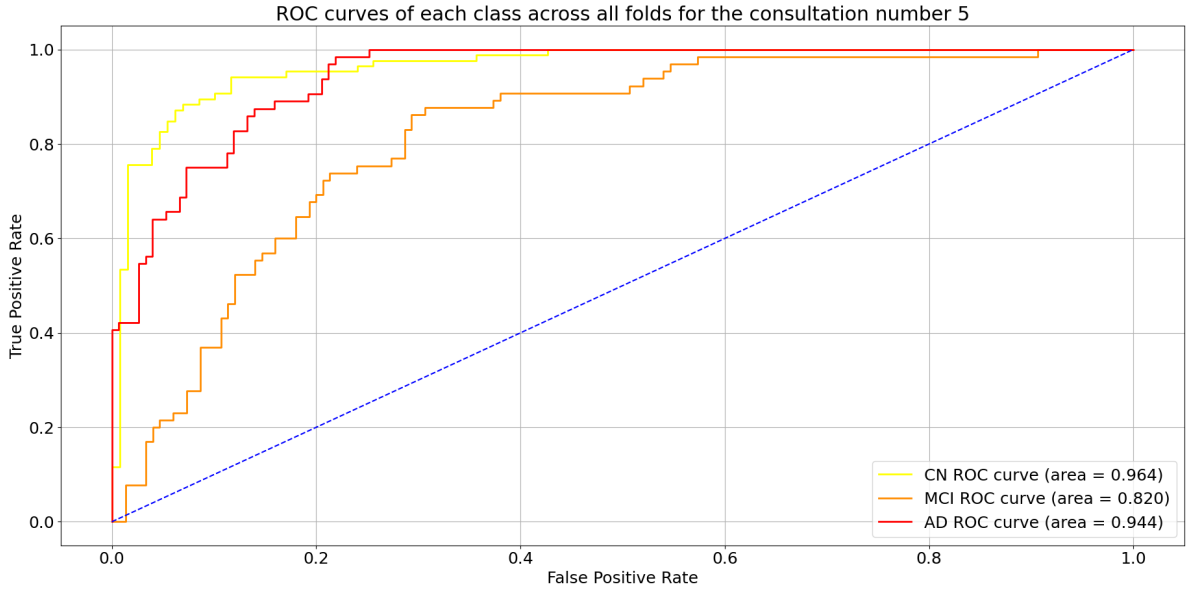

Figure A.3: ROC of the 3 classes at the 5th consultation for the mean imputed dataset.

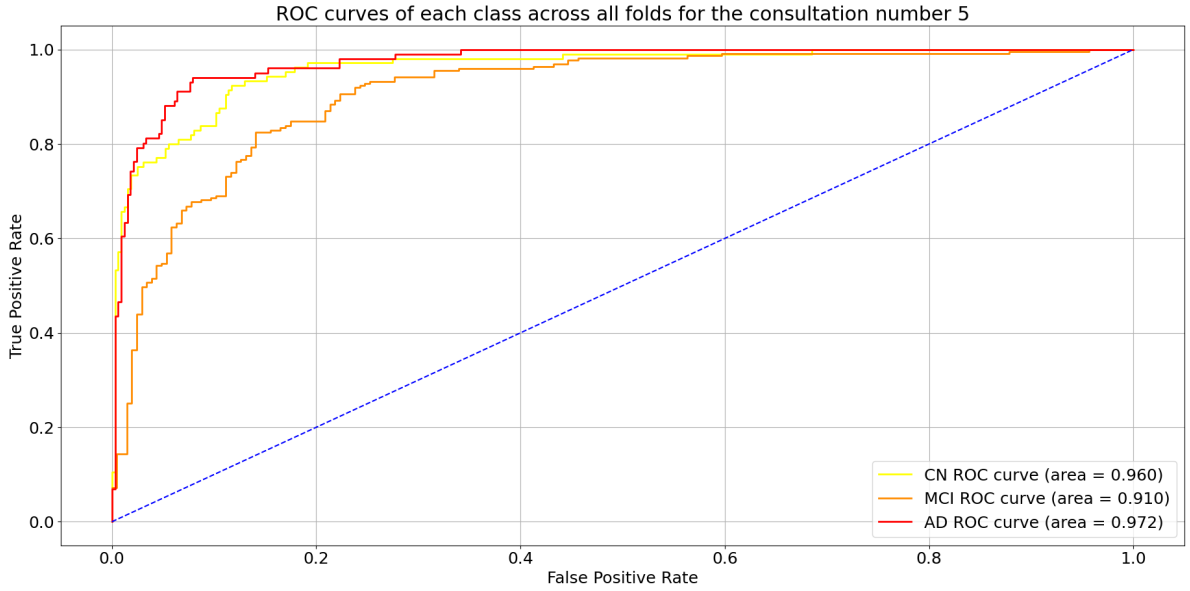

Figure A.4: ROC of the 3 classes at the 5th consultation for the masked dataset.

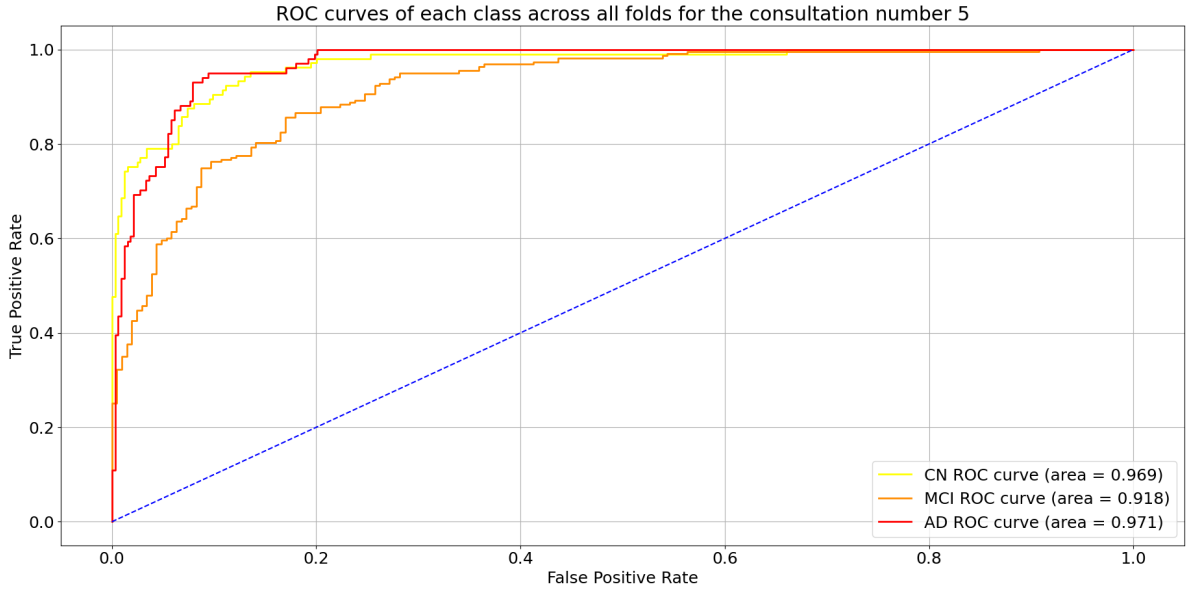

Figure A.5: ROC of the 3 classes at the 5th consultation for the GAIN imputed dataset.

Figure A.6: ROC of the 3 classes at the 5th consultation for the randomly imputed dataset.

ROC plots for the T-LSTM

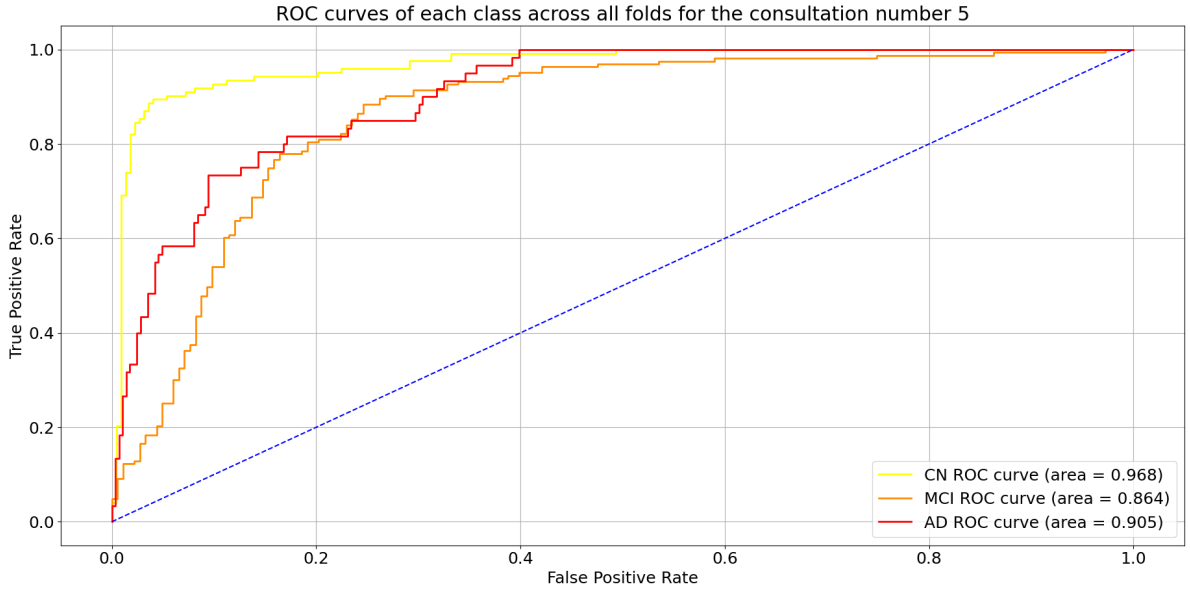

Figure A.7: ROC of the 3 classes at the 5th consultation for the full dataset.

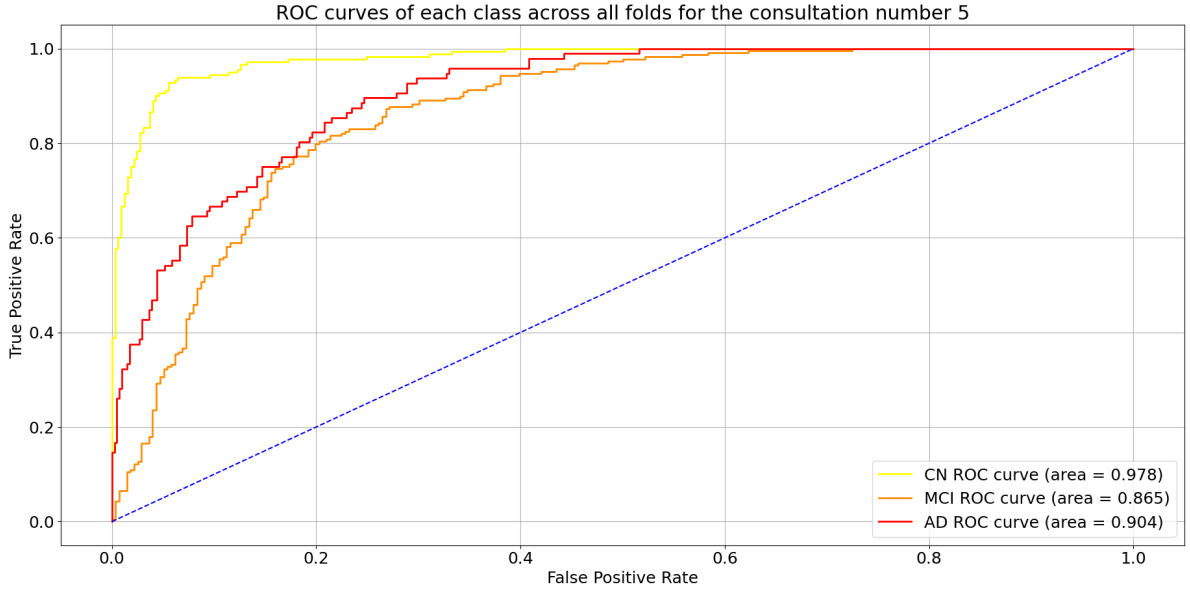

Figure A.8: ROC of the 3 classes at the 5th consultation for the mean imputed dataset.

Figure A.9: ROC of the 3 classes at the 5th consultation for the masked imputed dataset.

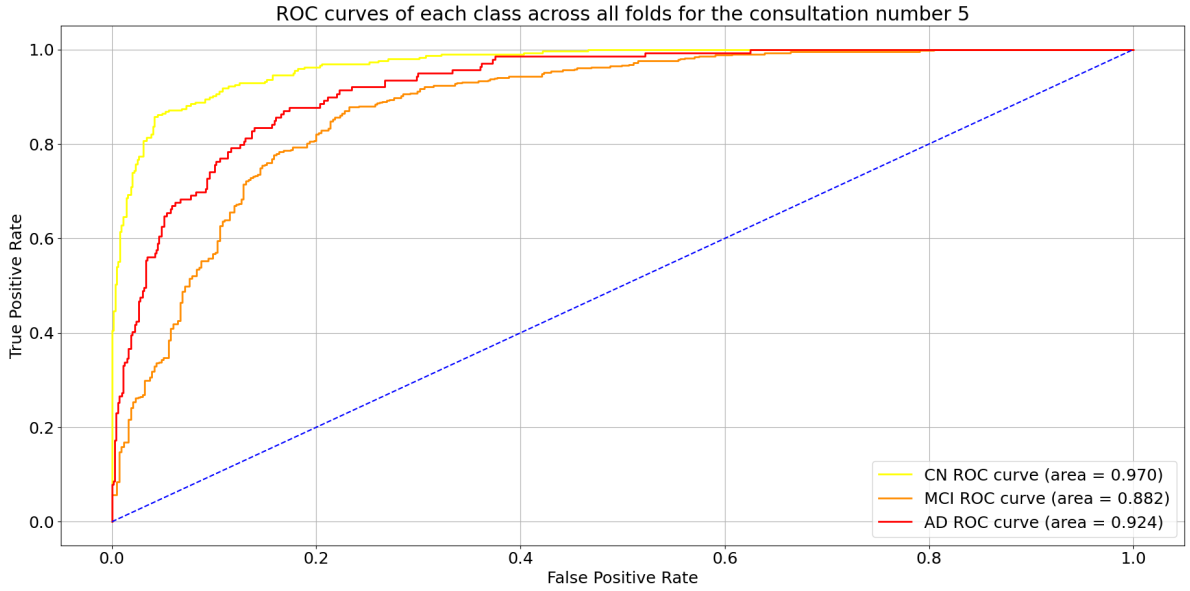

Figure A.10: ROC of the 3 classes at the 5th consultation for the randomly imputed dataset.

B

ROC of the Remaining Consultations

ROC plots for the LSTM

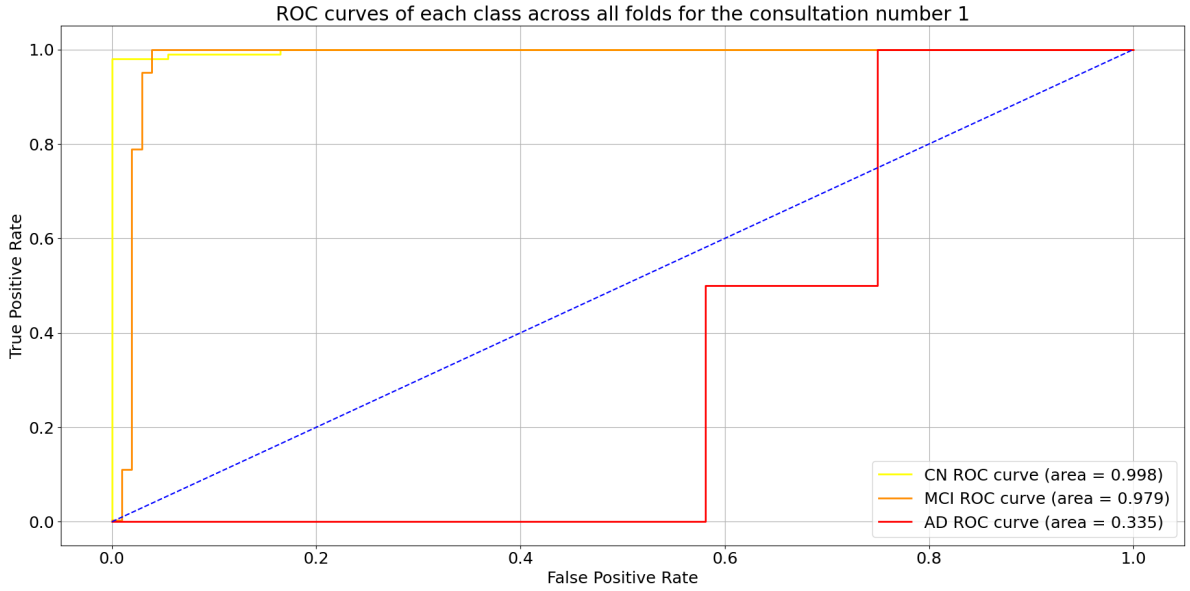

Figure B.1: ROC of the 3 classes at the 1st consultation for the VAE imputed dataset.

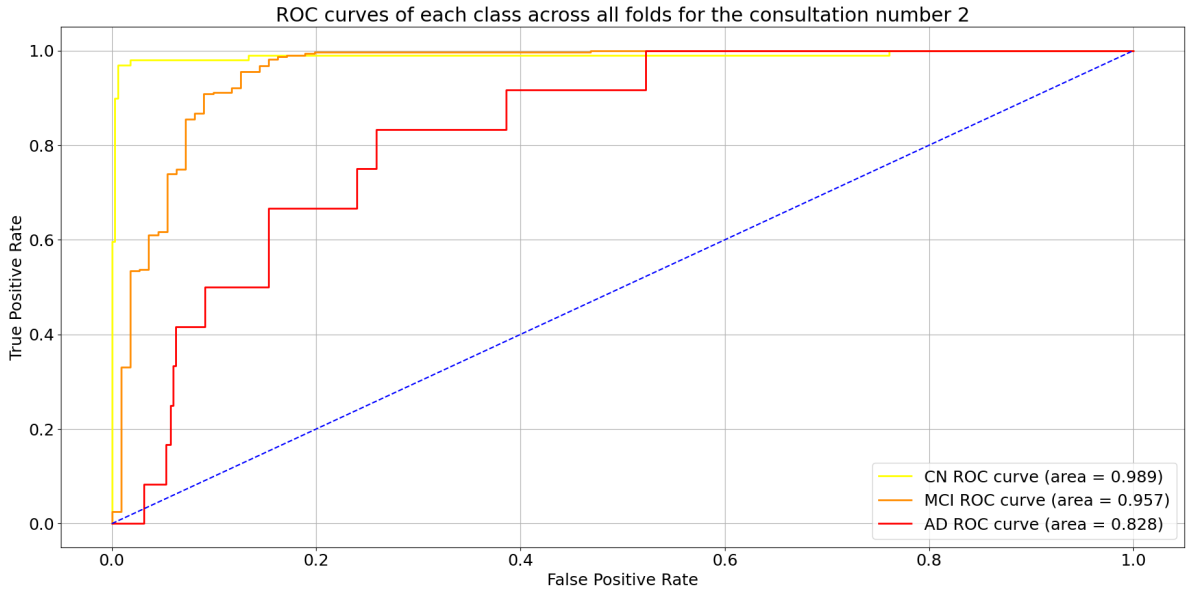

Figure B.2: ROC of the 3 classes at the 2nd consultation for the VAE imputed dataset.

Figure B.3: ROC of the 3 classes at the 3rd consultation for the VAE imputed dataset.

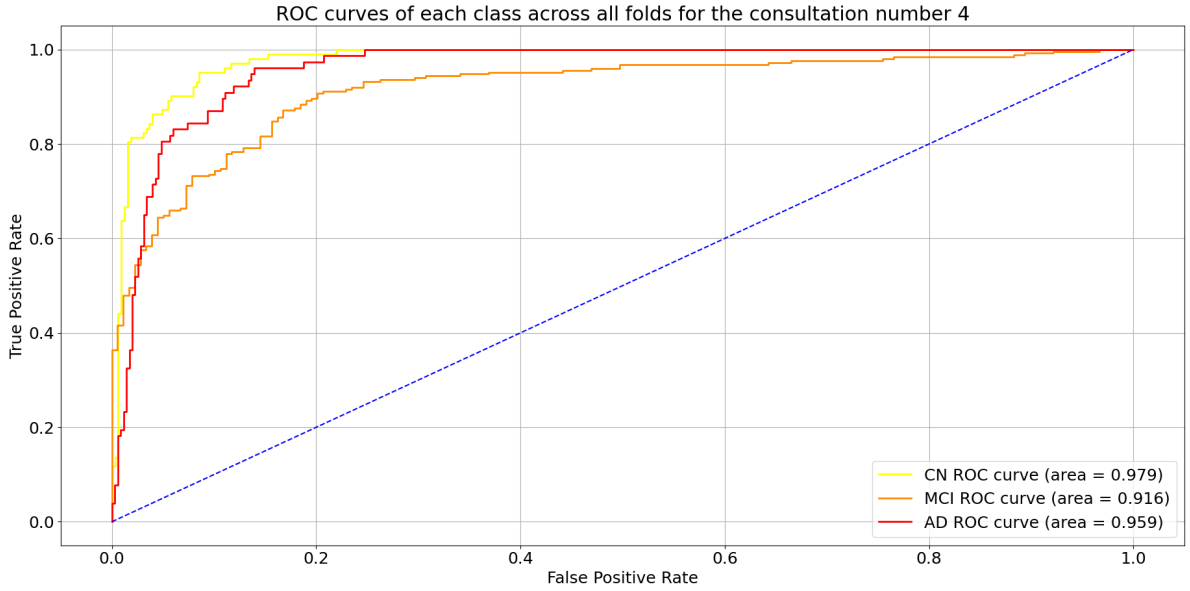

Figure B.4: ROC of the 3 classes at the 4th consultation for the VAE imputed dataset.

ROC plots for the T-LSTM

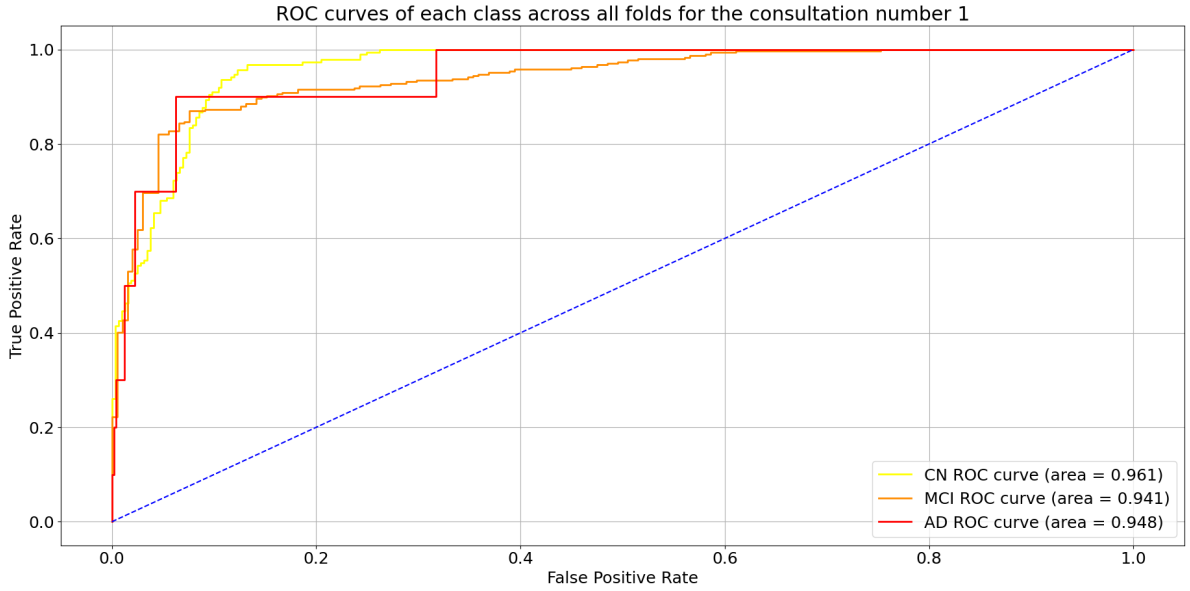

Figure B.5: ROC of the 3 classes at the 1st consultation for the Forward imputed dataset.

Figure B.6: ROC of the 3 classes at the 2nd consultation for the Forward imputed dataset.

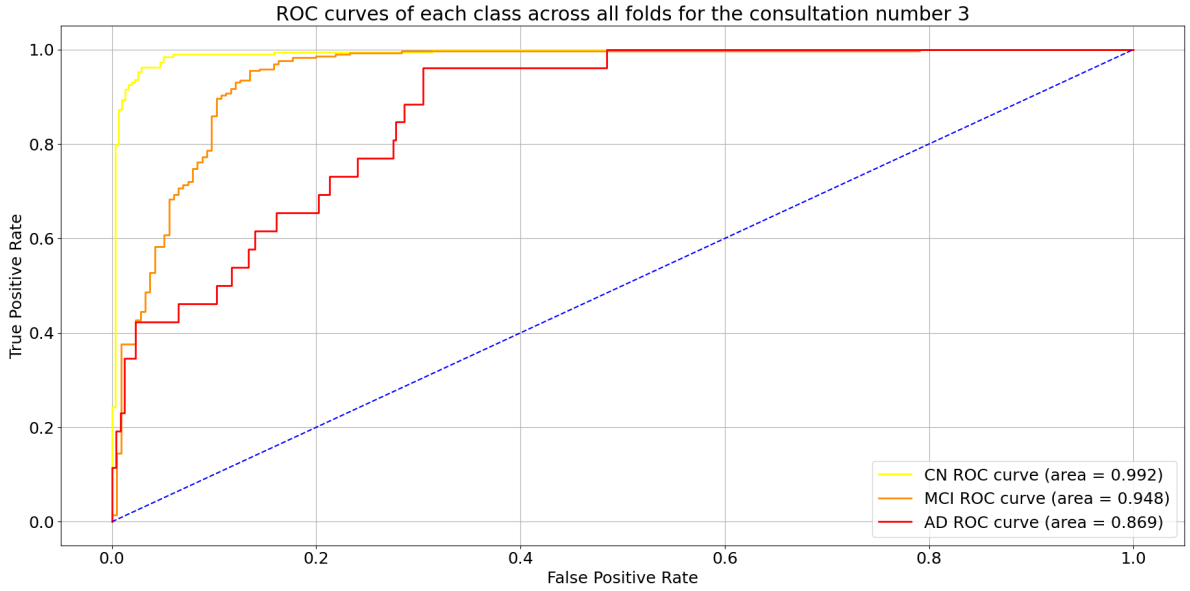

Figure B.7: ROC of the 3 classes at the 3rd consultation for the Forward imputed dataset.

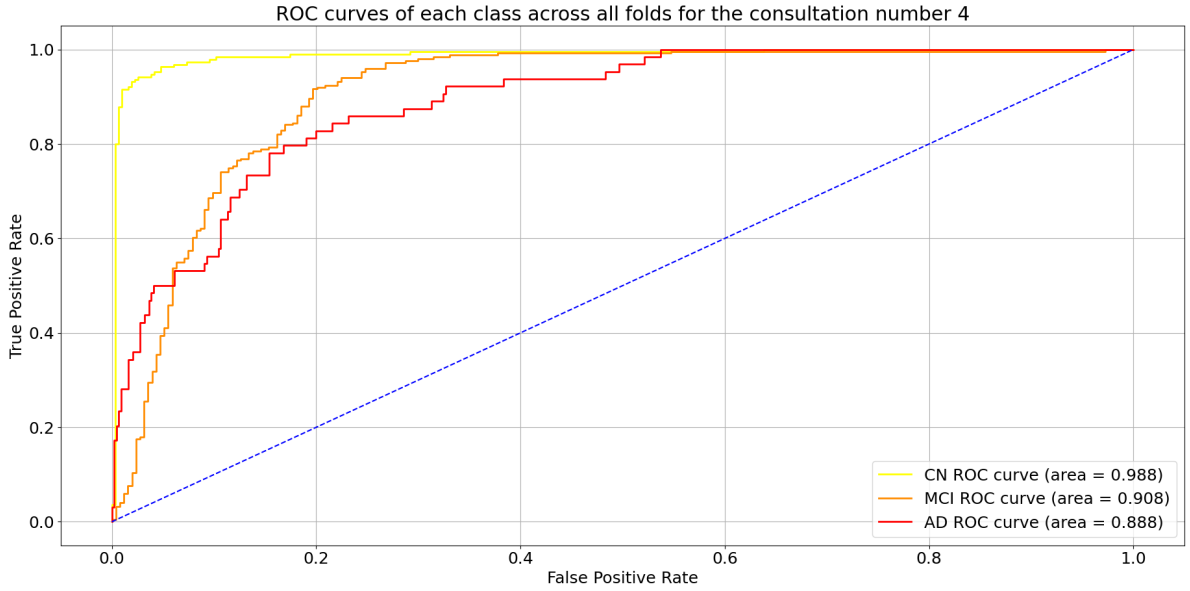

Figure B.8: ROC of the 3 classes at the 4th consultation for the Forward imputed dataset.