Alzheimer's Disease Onset Recognition by Handwriting: A Deep and Machine Learning Approach

Cartaya A.[1], Gonzalez J.[1], Hartmann F.[1], Rivas M.[1]

Alzheimer's disease is a neurodegenerative disorder characterized by cognitive impairments that progressively affect motor skills and cognitive abilities. Early diagnosis is crucial for slowing down brain damage and improving the quality of life for affected individuals. This study employs machine learning and deep learning approaches to analyze data from Alzheimer's patients, focusing on pen gesture dynamics and task-related images. Common deep learning architectures (VGG19, ResNet50, InceptionV3, and InceptionResNetV2) are utilized, and machine learning classifiers are tuned for optimal performance. Additionally, deep features are extracted from images to enhance analysis. It was confirmed that, by combining these approaches we can diversify the proper use of discriminant methods, to obtain better results depending on the type of written task.

1 Introduction

Alzheimer's disease is a prevalent neurodegenerative disorder worldwide, characterized by the progressive degeneration of nerve cells leading to cognitive impairments and motor skill decline. As the most common form of dementia, Alzheimer's disease affects millions of individuals and poses significant challenges to healthcare systems and society as a whole. Early diagnosis plays a crucial role in delaying disease progression and improving patient outcomes. However, accurately diagnosing Alzheimer's disease in its early stages remains a complex task.

In recent years, advancements in machine learning and deep learning techniques have shown promise in supporting the diagnosis and understanding of Alzheimer's disease. By leveraging the power of these computational approaches, researchers can analyze diverse data sources to uncover meaningful patterns and

Affiliation [1] Universita degli studi di Cassino e del Lazio Meridionale - UNICAS
Correspondence {agustinujarky.cartayalathulerie, jesusdavid.gonzalezriveros, frederik.hartmann, solangemicaela.rivasdiaz}@studentmail.unicas.it
Draft June 13, 2023

markers of the disease. One such approach is the analysis of handwriting dynamics, which has been identified as one of the early skills influenced by Alzheimer's disease onset.

According to the Alzheimer Organization [1] In Alzheimer's, the neurons damaged first are those in parts of the brain responsible for memory, language and thinking. As a result, the first symptoms tend to be memory, language and thinking problems. This leads to notable issues when simple motor tasks as speaking or writing, where patients struggle with following up with tasks as joining or keeping a conversation or a writing pattern.

Based on previous research, the target in this project is to combine different approaches in machine learning and deep learning methodologies to analyze data from Alzheimer's disease patients. The study focuses on two key aspects: the analysis of pen gesture dynamics during various tasks and the examination of the same task-related images using deep learning architectures. By integrating these approaches, a comprehensive understanding of the cognitive impairments associated with Alzheimer's disease can be achieved.

In addition to the deep learning analysis, the project will investigate the extraction of deep features from images. These deep features will then be utilized as inputs to machine learning classifiers, enabling a fusion of the information derived from both approaches.

The process is described as follows:

- Data Description and Acquisition
- Feature Selection and Preprocessing
- Classifier Selection and Hyperparameter tunning
- Deep Learning Approach
- Deep Features Extraction
- Results Analysis and Discussion

2 Dataset Description

Data is obtained by asking each participant to complete a series of tasks, divided in different categories and levels of difficulty, being the next task more demanding than the previous one in terms of the cognitive functions required to fulfill the task.

Overall, there are 25 tasks divided in three groups: Memory tasks (MT), copy and reverse copy tasks (CT) and graphic tasks (GT). All of them need to be written on A4 white sheets, which are stapled and placed on a graphic tablet which records the movements of the pen used by the examined subject [5].

The description of the tasks can be seen in 1 as explained in [5, 6]. For the scope of the project, a subset of the main tasks is utilized: tasks 1, 2, 3, 4 and 10.

Task 1 is a regular task used within the frame of written tasks for diagnosing cognitive impairments. Tasks 2 and 3 evaluate wrist joint movements and finger joint movements. Task 4 allows testing the automaticity of movements and the regularity and coordination of the sequence of movements[5, 6]. Task 9 allow testing the motion control alternation and Task 10 is a type of task designed to observe the variation of the spatial organization of the patient[5, 6].

As for the number of samples, the provided dataset, for both writing dynamics and images, consists of 166 observations, from which 88 belong to known affected patients, and 78 from the healthy control group.

In the writing dynamics, there are 92 features including ID of the patient, Sex, Age, the type of work the patient do for a living, as it is considered to be a subject of matter when diagnosing, the instruction level and the label: Healthy control or patient with Alzheimer's disease. The rest of the features describe the interaction of the pen while doing the tasks, based on observations while the pen is on the paper or in the air.

As described in [2] many studies in the literature show significant differences in patients' motor performance between in-air and on-paper traits. In-air refers to when the pen is lifted from the sheet.

Two types of features can be considered: function features and parameter features. The first characterize handwriting movements in terms of time functions, whereas the second are computed by means of a transformation upon the function features. The most common function features are: (x, y) coordinates, pressure, azimuth, altitude, displacement, velocity and acceleration. Some of these features are directly recorded by the acquisition device, whereas others are numerically derived. Typically, the most used function features are velocity and acceleration: the former contains information related to the slowness of movements, whereas acceleration changes allows tremor to be revealed. As for the features related to the in-air movements, it has been recently demonstrated that they convey very useful information for discriminating the movements of subject affected by Alzheimer's Disease.

Task#	Task Description	Task Type
1	Signature drawing	MT
2	Join two points with a horizontal line continuously for four times	GT
3	Join two points with a vertical line continuously for four times	GT
4	Retrace a 6cm-diameter circle continuously for four times	GT
5	Retrace a 3cm-diameter circle continuously for four times	GT
6	Copy the letters 'l', 'm' and 'p'	CT
7	Copy the letters 'n', 'l', 'o' and 'g' in adjacent rows	CT
8	write the letter 'l' 4 times, continuously in cursive format	CT
9	write the bigram 'le' 4 times, continuously in cursive format	CT
10	Copy the word "foglio"[1]	CT
11	Copy the word "foglio" above a line	CT
12	Copy the word "mamma"	CT
13	Copy the word "mamma" above a line	CT
14	Memorize the words "telefono", "cane" and "negozio"[2] and rewrite them	MT
15	Copy in reverse the word "bottiglia"[3]	CT
16	Copy in reverse the word "casa"[4]	CT
17	Copy six words	CT
18	Write of an object shown in a picture	MT
19	Copy the fields of a postal order	CT
20	Write a simple sentence under dictation	MT
21	Retrace a complex form	GT
22	Copy a telephone number	CT
23	Write a telephone number under dictation	MT
24	Draw a needle clock pointing 11:05	GT
25	Copy a paragraph	CT

Table 1: Tasks description and type [5, 6]

3 Feature Selection

3.1 Brief Description

In the previous section the dataset structure was described. In this section we present the subset selection techniques utilized to obtain different results.

The first natural approach was using the complete dataset for each of the selected tasks. This approach served as a first contact and exploratory analysis.

Then, based on the feature analysis done in [2] the approach used to select features was separating them into two groups: the ones reflecting the writing dynamics on-air (OA) and the ones doing it on-paper (OP). By visual inspection done with *ydata-profiling* package, it was seen that two of the features had no relevance at all (all values zero) so we removed those from all tasks. These features are:

- AveragePenPressureMeanOA
- AveragePenPressureStdOA

Both belonging to on-air dynamics (OA).

In this fashion, we have three datasets: The full dataset, the on-air version in which all OP features were removed and the on-paper version in which all OA features were removed. An important detail is that features related to the background of the participant are kept in all versions of the datasets.

3.2 Feature Selection

One of the best practices in machine learning, is to work with the most descriptive set of features of a given dataset. This will help reducing several issues that may be encountered dealing with data like noise or overfitting and improves the computational efficiency.

In order to find a set of features that will help in better discrimination between healthy and affected patients, we used recursive feature elimination (RFE) with cross validation (CV) and random forest (RF).

RFE performs a greedy search to and the best performing feature subset, based on the backward elimination strategy. Starting from the whole set of available features, the RFE algorithm iteratively creates models and determines the worst-performing feature at each iteration. Then, it builds the subsequent models with the features leftover until all the features are explored. If the data contain N features, in the worst case RFE evaluates N2 subsets. The algorithm provides as output the feature subset providing the best performance among those tested.

An important detail when working with RFE is that it requires an idea or criteria to start reducing dimensionality. That criteria in this case will be the feature importance, obtained after fitting the studied data into a Random Forest. After obtaining the importance of all features, RFE will be executed with proper CV in order to start the dimensionality reduction process.

As for the Random Forest, the number of trees of such forest is called number of estimators. In the case of the one used to reduce the dimensionality, were used two versions, one using 10 estimators and one with 400 estimators so we can compare if more estimators necessarily means better performance.

dataset	**dataset version**
Complete dataset with no FE	
Complete dataset with FE	10 RF estimators / 400 RF estimators
OA dataset with no FE	
OA dataset	10 RF estimators / 400 RF estimators
OP dataset with no FE	
OP dataset	10 RF estimators / 400 RF estimators

Table 2: Datasets used after feature extraction

This process will be done for each dataset: complete, OA and OP. In summary, we have the complete dataset without feature selection, the complete dataset after feature selection, the OA dataset after feature selection and the OP dataset after feature reduction, as indicated in 2.

3.3 Features per task

Since the task is different among each other, the number of features per task will vary according to the number of estimators used, being the final result the ones presented in 3 4 5:

task #	dataset	# of features
1	Complete dataset with no FE	88
	Complete dataset with FE - 10 RF estimators	17
	Complete dataset with FE - 400 RF estimators	77
	OA dataset with FE - 10 RF estimators	43
	OA dataset with FE - 400 RF estimators	8
	OP dataset with FE - 10 RF estimators	34
	OP dataset with FE - 400 RF estimators	26
2	Complete dataset with no FE	88
	Complete dataset with FE - 10 RF estimators	7
	Complete dataset with FE - 400 RF estimators	73
	OA dataset with FE - 10 RF estimators	21
	OA dataset with FE - 400 RF estimators	40
	OP dataset with FE - 10 RF estimators	16
	OP dataset with FE - 400 RF estimators	28

Table 3: features per dataset, tasks 1 and 2

task #	dataset	# of features
3	Complete dataset with no FE	88
	Complete dataset with FE - 10 RF estimators	2
	Complete dataset with FE - 400 RF estimators	2
	OA dataset with FE - 10 RF estimators	36
	OA dataset with FE - 400 RF estimators	15
	OP dataset with FE - 10 RF estimators	7
	OP dataset with FE - 400 RF estimators	11
4	Complete dataset with no FE	88
	Complete dataset with FE - 10 RF estimators	34
	Complete dataset with FE - 400 RF estimators	72
	OA dataset with FE - 10 RF estimators	40
	OA dataset with FE - 400 RF estimators	1
	OP dataset with FE - 10 RF estimators	26
	OP dataset with FE - 400 RF estimators	44

Table 4: features per dataset, tasks 3 and 4

task #	dataset	# of features
9	Complete dataset with no FE	88
	Complete dataset with FE - 10 RF estimators	32
	Complete dataset with FE - 400 RF estimators	5
	OA dataset with FE - 10 RF estimators	4
	OA dataset with FE - 400 RF estimators	19
	OP dataset with FE - 10 RF estimators	7
	OP dataset with FE - 400 RF estimators	16
10	Complete dataset with no FE	88
	Complete dataset with FE - 10 RF estimators	71
	Complete dataset with FE - 400 RF estimators	16
	OA dataset with FE - 10 RF estimators	3
	OA dataset with FE - 400 RF estimators	6
	OP dataset with FE - 10 RF estimators	45
	OP dataset with FE - 400 RF estimators	43

Table 5: features per dataset, tasks 9 and 10

Next step is defining the set of classifiers and the proper hyperparameters to tune in order to compare their performance.

4 Classifier Selection and Hyperparameter Tuning

In order to get a performance benchmark, it is necessary to test among different classifiers. For the sake of this project we used the following classifiers:

- Decision Tree classifier (DTC)
- Gradient Boosting Classifier (GBC)
- Linear Discriminant Analysis (LDA)
- Random Forest Classifier (RFC)

- Support Vector Machine (SVM)
- Extreme Gradient Boosting (XGB)

Is worth noting that Decission trees are baseline for random forest, as those forest are built of units of DTC. Another set of ensemble methods used are the boosting ones, from which Gradient Boosting Classifier and Extreme Gradient Boosting were selected. Support vector machines are widely known classifiers used in machine learning and LDA, aside of being a dimensionality reduction method, is itself a classifier.

Each of those classifiers have proper parameters involved in the way data is processed within. Each of this can be changed in order to achieve better results, and that is the process of hyperparameter tuning within the frame of k fold cross validation. The goal is, by fitting each classifier according to the size of the training and testing set, finding the best set of these parameter in order to get the best possible performance rate. Accuracy in this case.

A fundamental step when dealing with data is data preprocessing. Several classifiers benefit from having the data curated for the sake of fitting and delivering better results. This step is done inside the pipeline, so we are not limited to just one scaling type. Among scalers we tested standard, robust, minmax, maxabs, quartile, transformer and no scaler.

When preprocessing the data, outliers were also treated with different methods, namely z-score and its modified version, interquartile range and no detection. Depending of the position of the outlier, if any, the imputation will be done with either the 5 or the 95 percentile.

So, within a 5-fold cross validation as a subset selection strategy, the pipeline will actively search the best combination of parameters, scalers and outlier removal techniques. The hyperparameters of each of the classifiers are described in the table below:

Classifier	**Hyperparameters**	**Values tested**
Decision Tree	criterion	gini, entropy
	max depth	2,5,20, None
	min samples leaf	1, 5, 10
	max leaf nodes	2, 5, 10, 20
Boosting Classifier	# estimators	50, 100, 200
	learning$_rate$	0.1, 0.01, 0.001
	max depth	3, 5, 7
	min samples split	2, 5, 10
	min samples leaf	1, 2, 4
	max features	sqrt, log2
Linear Discriminant Analysis	solver	svd, lsqr, eigen
Random Forest	# estimators	100, 200
	max depth	10, 20, 50, 100, None
	min samples leaf	1, 2, 4
	min samples split	2, 5, 10
Support Vector Machine	kernel	linear, poly, rbf, sigmoid
	C	0.1, 1, 10, 100, 1000
	gamma	1, 1e-1, 1e-2, 1e-3, 1e-4
Extreme Gradient Boosting	min child weight	1, 5, 10
	gamma	0.5, 1, 1.5, 2, 2.5
	subsample	0.6, 0.8, 1
	col sample by tree	0.6, 0.8, 1
	max depth	3, 4, 5

Table 6: Classifiers and possible values used in hyperparameter tuning

5 Dynamic Features Selection Evaluation

Being selected the different set of features and the classifiers with the hyperparameters to be tuned, is a common question wonder if the feature selection indeed help to better discriminate between health and cognitive impaired patients. After running the complete fitting of each subset of features to each of the selected classifiers, we average the results of each of the scores obtained, using it as final value.

In order to summarize the results, they are organized in tables 7-12, using bold to highlight the best achieved results.

	OA			**OP**			**All dataset**		
Task	Full	RF10	RF400	Full	RF10	RF400	Full	RF10	RF400
1	68.6	69.2	69.8	68.7	69.8	**71.1**	69.2	68.7	69.2
2	68.7	65.7	62.8	69.8	71.7	**73.5**	69.3	70.5	69.3
3	68.7	66.3	65.1	67.5	**70.5**	68.7	66.9	69.9	69.9
4	68.7	63.9	68.6	67.5	68.7	69.2	68.7	**72.3**	68.6
9	68.1	58.9	68.1	68.1	**71.7**	68.1	68.1	64.4	68.1
10	68.7	72.9	**74.6**	68.7	60.3	68.6	68.7	61.51	68.6

Table 7: Classification results achieved by DTC

	OA			OP			All dataset		
Task	Full	RF10	RF400	Full	RF10	RF400	Full	RF10	RF400
1	70.5	65.7	69.9	74.2	**74.7**	68.7	72.3	66.9	72.3
2	72.9	**75.3**	72.3	70.5	71.7	72.9	71.1	74.7	71.1
3	69.9	70.5	73.5	73.4	74.1	**75.3**	71.7	73.4	68.0
4	68.1	64.5	67.4	70.4	65.6	**70.5**	70.5	69.3	69.9
9	65.0	60.1	67.4	72.3	65.0	**75.9**	66.9	66.2	69.8
10	68.0	65.0	75.3	71.0	66.9	70.5	69.9	65.0	**75.9**

Table 8: Classification results achieved by GBC

	OA			OP			All dataset		
Task	Full	RF10	RF400	Full	RF10	RF400	Full	RF10	RF400
1	66.7	69.1	66.9	69.3	62.1	**69.8**	59.6	62.1	64.5
2	62.0	65.0	62.0	65.6	66.3	**71.7**	60.2	69.9	62.7
3	63.9	69.4	69.3	59.6	**70.4**	68.0	53.6	70.4	70.4
4	63.2	62.1	68.0	64.5	63.9	67.6	63.3	**69.3**	68.7
9	62.6	65.0	64.5	63.9	69.3	68.7	59.6	63.3	**69.9**
10	57.3	69.9	65.7	65.0	69.3	64.4	51.2	61.4	**70.5**

Table 9: Classification results achieved by LDA

	OA			OP			All dataset		
Task	Full	RF10	RF400	Full	RF10	RF400	Full	RF10	RF400
1	69.8	69.9	72.9	73.5	73.5	**75.3**	71.7	74.7	71.6
2	72.3	74.1	72.3	68.6	71.0	71.7	71.1	**74.7**	72.3
3	70.5	66.9	71.7	72.2	74.0	**74.1**	71.1	69.8	70.4
4	68.1	66.3	67.4	70.5	71.1	71.1	**71.8**	69.3	71.7
9	63.2	69.2	68.0	68.7	68.7	**72.9**	66.2	67.4	69.2
10	68.0	66.2	**74.1**	71.7	69.8	71.0	70.4	67.5	73.5

Table 10: Classification results achieved by RFC

	OA			OP			All dataset		
Task	Full	RF10	RF400	Full	RF10	RF400	Full	RF10	RF400
1	66.9	71.7	70.5	72.9	71.0	**74.6**	65.6	72.3	69.2
2	68.1	68.7	71.1	72.9	69.2	**72.9**	69.9	70.5	71.1
3	66.9	65.7	71.0	71.0	**72.3**	72.2	69.2	72.2	72.2
4	66.9	71.1	71.6	68.7	68.1	69.3	69.3	**71.7**	68.6
9	67.4	67.4	**72.2**	69.2	69.9	70.5	66.8	66.9	69.2
10	67.4	68.1	71.6	72.3	67.4	**74.6**	66.3	64.4	71.7

Table 11: Classification results achieved by SVM

	OA			OP			All dataset		
Task	Full	RF10	RF400	Full	RF10	RF400	Full	RF10	RF400
1	**71.0**	70.5	68.0	64.5	67.4	66.3	68.7	70.5	69.9
2	60.2	61.4	60.2	59.0	63.8	63.2	63.3	**63.9**	63.3
3	62.1	66.3	65.7	69.3	68.6	**71.7**	67.5	70.4	70.4
4	59.6	64.4	60.8	64.5	**67.4**	62.0	60.2	63.8	63.2
9	62.6	60.1	63.8	66.9	62.0	**71.1**	64.4	68.0	60.8
10	63.8	66.2	65.1	66.8	**70.4**	67.4	65.7	67.4	69.3

Table 12: Classification results achieved by XGB

From the results we can select the best results per each task:

- Task 1: Random Forest Classifier on an on-paper dataset with features selected by a Random Forest of 400 estimators. The parameters to get these results are: max depth 20, min samples leaf 1, min samples split 200 and 200 estimators.
- Task 2: Gradient Boosting Classifier on an on-air dataset whose features were selected by a Random Forest with 10 estimators. The parameters to achieve these results are: learning rate 0.01, max depth 3, max features log2, min samples leaf 4, min samples split 10, and 200 estimators.
- Task 3: Gradient Boosting Classifier on an on-paper dataset whose features were selected by a Random Forest with 400 estimators. The parameters to achieve these results are: learning rate 0.01, max depth 3, max features log2, min samples leaf 2, min samples split 2, and 100 estimators.
- Task 4: Decision Tree classifier on a complete dataset (on-air and on-paper features) whose feature selection process was performed by a decision tree with 10 estimators. The parameters to achieve these results are: Criterion entropy, max depth 5, max leaf nodes 10, min samples leaf 5 and min samples split 2.
- Task 9: Gradient Boost Classifier on an on-paper dataset whose feature selection process was done by a Random Forest with 400 estimators. The

parameters to achieve these results are: learning rate 0.1, max depth 7, max features log2, min samples leaf 4, min samples split 2, and 100 estimators.

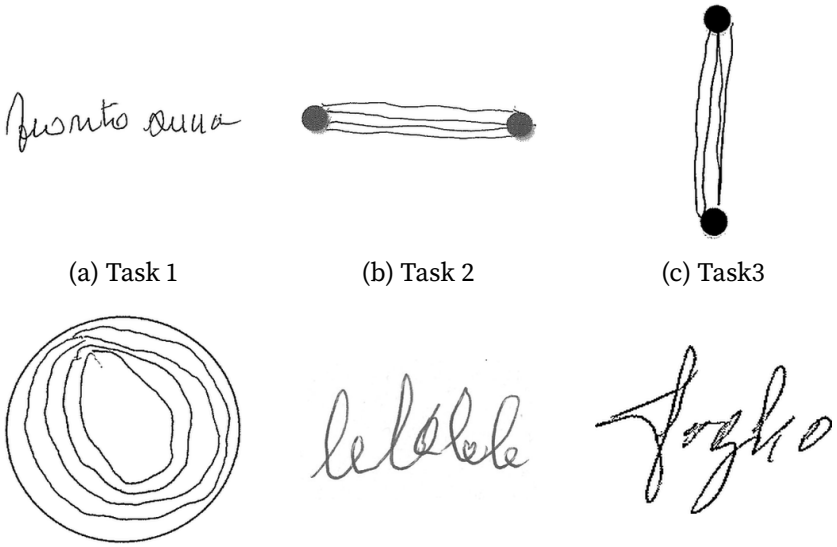
- Task 10: Gradient Boost Classifier on the complete dataset, whose features were selected using a Random Forest of 400 estimators. The parameters to achieve these results are: learning rate 0.1, max depth 5, max features sqrt, min samples leaf 2, min samples split 10, and 50 estimators.

There are several takeaways from this observations: The best results come from datasets that were modified by a feature selection process: four out of six task showed better performance using 400 estimators and the other two, using 10 estimators. Second takeaway is that Gradient Boosting Classifier outperform the selected methods, being just one instance in which the baseline decision tree show better result.

On paper dynamics tend to be more descriptive when it comes to distinguish among classes. 5 out of 6 tasks perfrom better with on paper tasks included, either pure on-paper or combined in all dataset along the on-air ones.

6 Deep Learning Approach

As mentioned before, part of the data obtained for this study, not only includes the dynamic features from pen movement, either on-air or on-paper. There is also the image of the pen traces for each of the tasks, obtained by the same protocol mentioned in [5]. All images are gray-scale images with size 229 by 229 pixels.



(a) Task 1 (b) Task 2 (c) Task3

(d) Task 4 (e) Task 9 (f) Task 10

Figure 1: Sample of one image per task

In order to analyze the images and use them as resources to automatically classify them between healthy or cognitive impaired, we will use convolutional neural networks (CNN) based on a transfer learning approach.

Transfer learning is a method used in deep learning (DL), that allows users to take advantage of a previously trained neural network and adapt it to the studied dataset. This can be achieved by freezing an amount of layers to keep the knowledge and generalization capacity previously learned. New introduced data will use this prior knowledge to train the unfrozen layers of the network, that generally are the classifying ones.

The overall process can be described as:

- Search for a pre-trained model
- Freeze the parameters in the convolutional layers. This step presents a trade-off between how much I want the network to learn from my dataset, without losing the existent generalization on known data.
- Add custom classifier layers on top of the base model.
- Train the new base plus custom classifier on your data.

- Tune the hyperparameters before freezing all again.

Right now is easy to find several convolutional neural networks trained on a lot of datasets. One of the most popular datasets is ImageNET [7], which has been used to train several base models. In this project we will use four different models: VGG19 [8], ResNET50[9], inceptionV3 [10] and inceptionResnetV2[11].

As briefly described above, next step would be decide the layers to unfreeze. So as part of the experiment, there will be trials between 0 and 3 convolutional layers. The next test will be evaluating different architectures of custom classifiers placed on top of the base CNN.

(a) CNN structure with droput layer

(b) CNN structure with dense layers on top

Figure 2: Custom Classifier architecture after base CNN model

Three custom classifier tests will be done, with the following structure:

- GlobalAveragePooling2D layer followed by 200-dense layer with ReLU as the activation function and one final 2-dense layer classifier based on sigmoid activation function. Following the b) structure in 2.
- GlobalAveragePooling2D layer followed by a 256-dense layer with ReLU activation function, then a 128-dense layer again with ReLU activation function, then a 64-dense layer with ReLU activation function and finally a 2-dense classifier layer using sigmoid as activation. Following the b) structure in 2.
- GlobalAveragePooling2D layer followed by a 256-dense layer with ReLU activation function, then a 0.5-dropout layer and then a 2-dense classifier layer with Softmax activation function. Following the a) structure in 2.

Finally, two optimization functions are to be tested. Adaptive Moment Estimation Optimazer, mostly known as *Adam* and Stochastic Gradient Descent (SGD).

We can set this as a "grid search" in order to find the best performance. Other required parameters for training convolutional networks are set as follows:

- Loss Function: Categorical Cross Entropy
- Batch Size: 10

Parameter	Values tested
Base CNN	VGG19
	inceptionResnetV2 (ir2)
	Inception V3
	Resnet50
Optimizer	Adam
	SGD
Layers to Unfreeze	[0, 1, 2, 3]

Table 13: Handmade gridsearch for DL approach

- Epochs: 10
- Steps per epoch: is given by the formula $Trainingdatasamples/batchsize$
- Training metric: accuracy

6.1 PHASE 1

Having all set, a handmade gridearch 13 was done with 32 possible candidates per task.

One important matter is that, even if is a good practice to augment data when datasets are small (evaluated on an application basis), that was not the case in this project. Previous experiments done on this dataset, showed that performance overall was better if no data augmentation was done. With this in mind, the data handling approach was a random subset for train and validation phases.

After running this first experiment, the top 4 results of each task are selected, based on the area under the sensitivity and specificity curve, mostly known as receiver operator caracteristic curve, ROC. Other useful metrics are also calculated: sensitivity, specificity, precision, f1-score, AUC, Matthew's correlation coefficient (MCC) and balanced accuracy (BA).

	Model	Optimizer	Unfrozen layers	Acc	Sen	Spe	Pre	F1	AUC	MCC	BA
29	VGG19	Adam	2	64.0	59.2	69.5	69.5	64.0	0.72	0.28	64.4
22	inceptionv3	SGD	3	60.0	77.7	39.1	60.0	67.7	0.69	0.18	58.4
3	resnet50	Adam	1	62.0	59.2	65.2	66.6	62.7	0.62	0.24	62.2
14	iR2	SGD	3	58.0	66.6	47.8	60.0	63.1	0.61	0.14	57.2
17	inceptionv3	Adam	0	68.0	92.5	39.1	64.1	75.7	0.84	0.38	65.8
10	iR2	SGD	1	72.0	85.1	56.5	69.7	76.6	0.84	0.43	70.8
1	resnet50	Adam	0	72.0	96.2	43.4	66.6	78.7	0.82	0.47	69.8
30	VGG19	SGD	3	68.0	62.9	73.9	73.9	68.0	0.81	0.36	68.4
29	VGG19	Adam	2	84.0	77.7	91.3	91.3	84.0	0.89	0.69	84.5
7	resnet50	Adam	3	68.0	44.4	95.6	92.3	60.0	0.84	0.45	70.0
14	ir2	SGD	3	74.0	77.7	69.5	75.0	76.3	0.82	0.47	73.6
16	inceptionv3	SGD	0	74.0	92.5	52.1	69.4	79.3	0.81	0.49	72.3

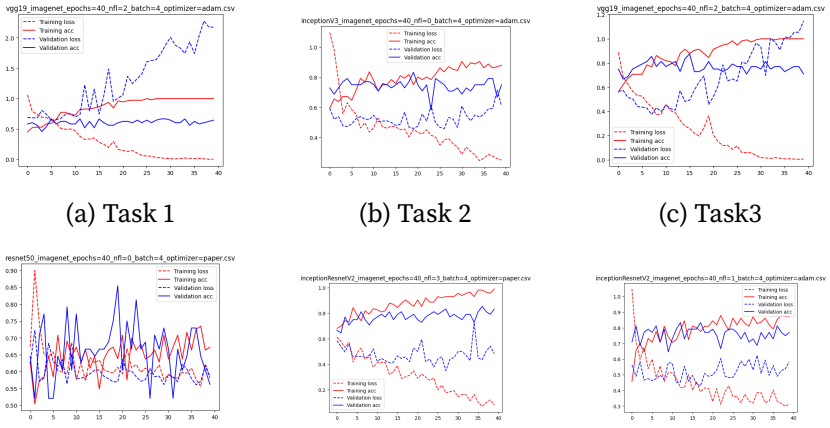
Table 14: CNN results for tasks 1, 2 and 3

	Model	Optimizer	Unfrozen layers	Acc	Sen	Spe	Pre	F1	AUC	MCC	BA
29	resnet50	SGD	0	77.0	60.0	95.6	93.7	73.1	0.90	0.58	77.8
22	inceptionv3	Adam	0	70.0	80.0	60.8	68.9	74.0	0.85	0.41	70.4
3	ir2	SGD	0	68.7	56.0	82.6	77.7	65.1	0.82	0.39	69.3
14	VGG19	Adam	0	70.8	48.0	95.6	92.3	63.1	0.80	0.49	71.8
17	iR2	Adam	3	70.0	48.1	95.6	92.8	63.4	0.88	0.48	71.9
10	inceptionv3	SGD	2	72.0	74.0	69.5	74.0	74.0	0.82	0.43	71.8
1	VGG19	Adam	2	74.0	70.3	78.2	79.1	74.5	0.78	0.48	74.3
30	resnet50	SGD	1	68.0	48.1	91.3	86.6	61.9	0.65	0.42	69.7
29	ir2	Adam	1	84.0	81.4	86.9	88.0	84.6	0.89	0.68	84.2
7	VGG19	Adam	2	78.0	70.3	86.9	86.3	77.5	0.88	0.57	78.6
14	inceptionv3	SGD	3	86.0	85.1	86.9	88.4	86.7	0.88	0.71	86.0
16	resnet50	Adam	3	70.0	96.2	39.1	65.0	77.6	0.74	0.44	67.7

Table 15: CNN results for tasks 4, 9 and 10

6.2 PHASE 2

After this initial step, the top 2 of the top 4 best performers of each task, were selected to do additional tests and improve the performance achieved so far. They were retrained with more epochs than the initial training. The overall performance of this second phase can be shown in the following images, were training and validation loss and accuracies can be seen per epoch of training in 3.



(a) Task 1 (b) Task 2 (c) Task3

(d) Task 4 (e) Task 9 (f) Task 10

Figure 3: Training and loss per training and validation for each task

6.3 PHASE 3

An additional last step of experimenting is added. In this one, an ensemble method is proposed, joining the best performers for each task. Results were improved in tasks 1 and 2, compared to the initial training and the improved training phases. The final selected models ended up being:

- Task 1: Phase 3 Ensemble of VGG19 + inceptionv3 for an AUC of 0.73, improving the result in 14
- Task 2: Phase 3 Ensemble of inceptionV3 + inceptionResnetV2 with an AUC of 0.85 improving the result in 14.
- Task 3: Phase 2 VGG19 with an AUC of 0.90 improving the result in 14.
- Task 4: Phase 1 resnet50 with an AUC of 0.90.

- Task 9: Phase 2 inceptionResnetV2 with an AUC of 0.89 improving the result in 15
- Task 10: Phase 1 inceptionResnetV2.

7 Deep Features Extraction

As a final step, following the research in [3], an advanced image analysis method is used, in which, features proper of image analysis, are extracted using convolutional neural networks. Once the image passes through a set of convolutional layers, descriptive image features are selected and stored for further machine learning analysis.

The data, coming from 100 image features, is then treated with the same procedures used in section 5: A pipeline including gridsearch for hyperparameter tuning within a frame of k-fold cross validation.

Classifiers selected in this section are: Random forest classifier, Decission Tree, Support Vector Machine, Extreme Gradient Boosting and Multi-layer Perceptron (MLP)

Task			**1**		
Classifier	**RFC**	**DTC**	**SVM**	**XGB**	**MLP**
Accuracy	0.58	**0.66**	0.62	0.62	0.54
F1-Score	0.618	0.666	0.612	0.641	0.701
Precision	0.607	0.708	0.681	0.653	0.54
Recall	0.629	0.629	0.555	0.629	1.0
Task			**2**		
Classifier	**RFC**	**DTC**	**SVM**	**XGB**	**MLP**
Accuracy	0.72	0.64	**0.76**	0.7	0.46
F1-Score	0.758	0.639	0.777	0.716	0.0
Precision	0.709	0.695	0.777	0.730	0.0
Recall	0.814	0.592	0.777	0.703	0.0
Task			**3**		
Classifier	**RFT**	**DTC**	**SVM**	**XGB**	**MLP**
Accuracy	0.8	0.8	**0.84**	0.82	0.76
F1-Score	0.814	0.799	0.84	0.823	0.799
Precision	0.814	0.869	0.913	0.875	0.727
Recall	0.814	0.740	0.777	0.777	0.888

Table 16: Performance Achieved using Image Features. Tasks 1, 2, 3

Task			**4**		
Classifier	**RFT**	**DTC**	**SVM**	**XGB**	**MLP**
Accuracy	0.75	0.770	0.729	**0.833**	0.520
F1-Score	0.777	0.8	0.648	0.846	0.684
Precision	0.724	0.733	1.0	0.814	0.520
Recall	0.84	0.88	0.48	0.88	1.0
Task			**9**		
Classifier	**RFT**	**DTC**	**SVM**	**XGB**	**MLP**
Accuracy	**0.8**	0.74	0.76	0.78	0.54
F1-Score	0.807	0.734	0.75	0.784	0.701
Precision	0.84	0.818	0.857	0.833	0.54
Recall	0.777	0.666	0.666	0.740	1.0
Task			**10**		
Classifier	**RFT**	**DTC**	**SVM**	**XGB**	**MLP**
Accuracy	0.78	0.7	0.7	**0.8**	0.54
F1-Score	0.799	0.716	0.736	0.827	0.701
Precision	0.785	0.730	0.7	0.774	0.54
Recall	0.814	0.703	0.777	0.888	1.0

Table 17: Performance Achieved using Image Features. Tasks 4, 9, 10

8 Test Result and Conclusion

Among the selected tasks for this project, one of the more complex appears to be the task 1 due to how individual are each of the samples, not being possible to compare the outcomes. N observations mean N complete different samples, not being possible to graphically match one sample from a healthy patient with one cognitive impaired patient.

This hypothesis could be the reason why in task 1, the analysis of the writing

dynamics outperform the image analysis with the deep learning approach, also, the deep learning feature analysis results may not present a good result. However, present an improvement with respect to [3].

Another interesting detail is that as time passes by, the signature, as a common activity in the human can lead individuals to memorize the pattern. This may cause signing as a motor activity.

This is not the case for the other copy tasks, where repetitive patterns belong to the form of the written trace in both healthy control group and cognitive impaired patients. Tasks 9 and 10 have the best results using the deep learning approach as discriminant. Deep learning features also present a good result comparing to the dynamic features performance.

As per the graphic tasks, deep learning features appear to perform better, and for task 2, the writing dynamic features seem to better discriminate between patients.

In this fashion, would not be naive at all to say that depending on the type of written task, one classifier would be more beneficial one approach or the other. Copy tasks with unique forms like the signature, benefit from the dynamics of the writing, considering on-air and on paper features.

Graphic tasks tend to perform better with deep learning features and copy tasks, on the other hand perform better with the use of deep learning.

One important aspect is the fact that data augmentation was not used. Beyond performing better when not using it, it would be pointless to distort patterns that need to remain immutable at the moment of the analysis. One "bad" linear transformation will definitely lead to a misclassified output.

References

[1] Alzheimer's Association. "2023 Alzheimer's Disease Facts and Figures." in Alzheimers Dement 2023;19(4). DOI 10.1002/alz.13016. https://www.alz.org/media/Documents/alzheimers-facts-and-figures.pdf

[2] N. D. Cilia, C. De Stefano, F. Fontanella and A. S. D. Freca, "Feature Selection as a Tool to Support the Diagnosis of Cognitive Impairments Through Handwriting Analysis," in IEEE Access, vol. 9, pp. 78226-78240, 2021, doi: 10.1109/ACCESS.2021.3083176. https://ieeexplore.ieee.org/document/9439485

[3] N. Dalia Cilia, T. D'Alessandro, C. De Stefano and F. Fontanella, "Offline handwriting image analysis to predict Alzheimer's disease via deep learning," 2022 26th International Conference on Pattern Recognition (ICPR), Mon-

treal, QC, Canada, 2022, pp. 2807-2813, doi: 10.1109/ICPR56361.2022.9956359. https://ieeexplore.ieee.org/document/9956359

[4] N. D. Cilia, G. De Gregorio, C. De Stefano, F. Fontanella, A. Marcelli, A. Parziale, "Diagnosing Alzheimer's disease from on-line handwriting: A novel dataset and performance benchmarking", Engineering Applications of Artificial Intelligence, Volume 111, 2022, 104822, ISSN 0952-1976, https://doi.org/10.1016/j.engappai.2022.104822.https://www.sciencedirect.com/science/article/pii/S0952197622000902

[5] [1] N. D. Cilia, C. De Stefano, F. Fontanella, A. S. Di Freca, "An experimental protocol to support cognitive impairment diagnosis by using handwriting analysis", Procedia Computer Science 141 (2018) 466–471. https://doi.org/10.1016/j.procs.2018.10.141

[6] N. D. Cilia, G. De Gregorio, C. De Stefano, F. Fontanella, A. Marcelli, A. Parziale, "Diagnosing Alzheimer's disease from online handwriting: A novel dataset and performance benchmarking", Engineering Applications of Artificial Intelligence, Vol. 111 (20229) 104822. https://doi.org/10.1016/j.engappai.2022.104822

[7] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li and L. Fei-Fei, "ImageNet: A Large-Scale Hierarchical Image Database". IEEE Computer Vision and Pattern Recognition (CVPR), 2009.

[8] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," CoRR, vol. abs/1409.1556, 2015.

[9] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770–778, 2016.

[10] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2818–2826, 2016.

[11] C. Szegedy, S. Ioffe, and V. Vanhoucke, "Inception-v4, inception-resnet and the impact of residual connections on learning," in AAAI, 2016.