Semi-supervised methods for the Diagnosis of Alzheimer's Disease from 3D Brain images

Frederico Figueira Alcobia

Department of Electrical and Computer Engineering
Instituto Superior Técnico
Av. Rovisco Pais, 1049-001 Lisboa, Portugal

Abstract— Currently with no cure, Alzheimer’s disease (AD) is the most common type of dementia, accounting for up to 80% of the world's registered cases of dementia. Given the increasingly significant social and economic impact of AD, its early detection is then a question of great importance, since it may contribute to the deceleration of its progress. Presently, multiple machine learning techniques are already implemented in the detection and classification of both AD patients, and patients with mild cognitive impairment that may, or may not, develop AD (labeled MCI-C and MCI-nC respectively), with satisfactory results. These techniques are usually based on supervised learning methods, methods that require data, that is normally obtained through Neuroimaging techniques such as Positron Emission Tomography (FDG-PET), that must have been previously labeled. The labeling process is expensive, in the sense that it requires the assessment, knowledge, and time of a specialist. Then comes the opportunity to exploit semi-supervised learning methods, methods that do not require or so heavily depend on the availability of classified data, while also benefiting from any supply of non-classified data. In this thesis, two semi-supervised methods based on Support Vector Machines (SVM) are presented and tested: Self-training (ST) and Co-training (CT). Their integration into computer aided diagnosis (CAD) automated systems is analyzed, for the classification of patients in: AD vs. normal control (NC) and MCI-C vs MCI-nC. The results obtained were comparable to those achieved by the supervised systems found in the literature but are much less dependent on the availability of training data and on the subjective labeling by the medical doctors.

Index Terms — Alzheimer’s disease, Mild Cognitive Impairment, Positron Emission Tomography, Semi-supervised Learning, Computer aided diagnosis, Self-training, Co-training

I. INTRODUCTION

In 2013, 44.4 million cases of dementia were registered worldwide [8], with the number of cases showing a tendency to double every 20 years. Out of those cases, Alzheimer's Disease (AD) is known to be the most common type of dementia, accounting for up to 80% of the world's dementia patients [11]. Taking that into consideration, and the fact that the costs associated with dementia are now over 500 thousand million euros every year [10], the economic impact of AD can then be said to be an ever-increasing problem. Moreover, with the ageing of the population, AD is becoming one of the major causes of death worldwide, sitting, as of the end of 2013, in sixth place on the list of causes of death in the USA, and showing a tendency for the number to increase, as opposed to diseases like HIV and cancer [9]. Presently with no cure, the early detection of AD is then a question of great importance.

Mild Cognitive Impairment (MCI) is an intermediate stage between the expected cognitive decline of normal ageing and the more serious decline of dementia [12]. Even though MCI increases the chances of developing AD or other dementias, not everyone affected by this problem ends up doing so. There is then a great interest in developing new diagnostic techniques that are able to distinguish between those who will end up developing AD, labeled MCI converters (MCI-C), and those who will remain stable, labeled MCI non-converters (MCI-nC). Neuroimaging techniques like the Positron emission tomography (PET) and Magnetic resonance imaging (MRI) have been recognized as powerful tools in the detection of AD. These techniques try and search for evidence specifically associated with the development of AD. Based on these techniques, Computer Aided Diagnostic (CAD) systems have been built in order to help physicians process the vast quantities of information neuroimaging techniques produce, thus helping them to classify patients.

A. Supervised vs. Semi-supervised Learning

Supervised learning is the machine-learning task of inferring a function from labeled training data, or examples, consisting of an input object, usually a feature vector, and a corresponding label. That function can then be used to map new data. By increasing the number of labeled examples, the resulting function can then better learn about the attributes linked to a certain label and thus increase the classification performance of new data. However, the acquisition of labeled data often requires a skilled human agent, or in the case of neuroimaging classification, a medical specialist, making it expensive to gather new examples.

On the other hand, unlabeled data is relatively inexpensive and usually much more abundant. Trying to capitalize on that, semi-supervised learning considers both labeled and unlabeled data when trying to define the classification function.

![Figure 1](image)

Figure 1 - Considering the unlabeled data may lead to a decision boundary with a better generalization ability, as seen in situation 2), compared to 1) that only considers the provided labeled examples.
B. Proposed Approach

The present work focused on the implementation of two semi-supervised methods, Self-training (ST) and Co-training (CT), in CAD systems made for the classification of AD. The CAD systems developed were built upon FDG-PET images, a specific kind of PET scan that will be introduced in chapter II, and were tested as semi-supervised alternatives to the existing supervised systems available. Specifically, they were applied to the classification problem of trying to distinguish between: AD vs. Healthy subjects/Normal control (NC), and MCI-C vs. MCI-nC.

The semi-supervised algorithms presented in this work were adapted to the standard CAD system architecture, that is usually made up of four steps: the image acquisition step, followed by feature extraction and selection so that the last step, the classification, can be performed. One of the most frequently used binary classifier and also the one implemented in this work is the Support Vector Machine (SVM) classifier.

![Figure 2 - CAD system architecture](image)

C. State of the Art

Ever since the beginning of the 21st century, many CAD systems have been created with the purpose of developing tools able to help physicians identify and label brain images produced by diverse neuroimaging techniques such as MRI [1, 2], PET [3, 4, 5], and Single-photon emission computed tomography (SPECT) [6, 7].

In these systems, the features obtained from the neuroimages play a major role in all the steps and calculations made afterwards. Those features can be obtained by a variety of methods: that can use the whole brain, like voxel intensity (VI) [3, 6, 7], or methods that try and concentrate on specific brain regions, associated with the development of AD, like regions of interest (ROIs) [1, 2, 5].

For the step of feature selection, with the goal of reducing the curse of dimensionality (small sized sets of data with large numbers of features) many algorithms have also been implemented. From methods that find linear combinations of features like Principal Component Analysis (PCA), Linear Discriminant Analysis (LDA) [7] or Non-negative Matrix Factorization (NMF) [6], to ranking algorithms that assign a weight to each feature according to their relevance, like the Fisher Discriminant Ratio (FDR) [6], or the Pearson Correlation Coefficient [4], these methods play a key role in trying to reduce the dimensionality of the input in the CAD systems.

The final step of classification is most frequently done using the learning algorithm of SVM, which was also used in the current work. Besides SVM, the other common factor between the majority of the systems developed in the past, is that supervised learning was almost always the method utilized for the classification step.

II. METHODS

As mentioned before, the semi-supervised CAD systems developed in the present work follow the architecture presented in figure 2:

A) Image Acquisition: In this work FDG-PET images from the ADNI [13] database were used;
B) Feature Extraction: the Voxel Intensity method was employed in this step;
C) Feature Selection: this step was done by calculating the Mutual Information between each feature of the FDG-PETs and the label vector and selecting the features with highest mutual information;
D) Classification: finally, in this last step, the two semi-supervised algorithms proposed in this work, ST and CT, based on SVM, label the elements accordingly, as: AD, MCI-C, MCI-nC or NC.

A. Image Acquisition

Alzheimer’s Disease Neuroimaging Initiative (ADNI) is a Public-Private Partnership, formed in 2004, that unites researchers with study data as they work to define the progression of AD [14]. ADNI researchers collect, validate and utilize data such as MRI and PET images, genetics, cognitive tests and blood biomarkers as predictors for the disease. They gather data not only from subjects suffering from AD but also from MCI and NC patients. In this work, an FDG-PET database from ADNI was used.

1) FDG-PET

The Neuroimaging technique utilized in this work was the fludeoxyglucose based, positron emission tomography, a nuclear medicine technique that produces a three-dimensional (3D) image of functional body processes based on the detection of pairs of gamma rays emitted by a positron-emitting radionuclide which is introduced into the body on a biologically active molecule [15]. This kind of PET utilizes FDG, an analogue of glucose, as the biologically active molecule. The scan then produces an image that measures the regional glucose uptake, allowing for the detection of one of the early-sings associated with AD: the reduction of metabolism in certain areas of the brain.

The resulting 3D FDG-PET volumes used in this work were comprised of 60 slices of the brain, with 128 by 128 voxels in each slice, amounting to 983040 total features for any given patient.

![Figure 3 - PET images showing the state of glucose consumption in the brain. Source: UCLA Longevity Center.](image)

2) Applied Database

As mentioned before, an FDG-PET database provided by ADNI was exploited in the current work. The composition of that database is presented in table 1.
The balance in the number of elements from each class is not random and was something the present work tried to maintain throughout all the steps of the suggested algorithms.

The provided database had been preprocessed in order to convert the data to a common format and orientation. In this work, the main components utilized from the database were:

- X (features): includes the FDG-PET images of the brain for all 896 patients, in form of a vector with the 983040 features from each patient;
- Y: contains the labels associated with each of the 896 FDG-PET.

B. Feature Extraction - Voxel Intensity

Voxel Intensity (VI) is a commonly used feature extraction method in 3D imaging. With it, features were obtained directly from the PET scan and their value \( I(x, y, z) \), where \( 1 \leq x \leq 128 \) and \( 1 \leq y \leq 128 \) and \( 1 \leq z \leq 60 \), when considering FDG-PETs, is a direct measure of the FDG uptake detected in a certain voxel.

Trying to reduce the dimensionality of the input, one more step was performed before feature the selection step. Its goal was to ignore all the voxels that lie outside the brain. In order to achieve that, an average of all 983040 voxel’s intensities was calculated, with a threshold then being set for which all voxels with lower values were discarded. After some observational adjustments that value was set to 4.1 % of the maximum registered value, resulting in 309338 final features.

C. Feature Selection - Mutual Information

Mutual Information (IM) is a measure of statistical dependency. IM measures the level of information a random variable provides about another random variable. Intuitively, IM measures how much knowing one of these variables reduces uncertainty about the other. Shannon introduced the concept in 1948 [16]. In its bivariate form, and given two random variables \( X \) and \( Y \), IM is the Kullback-Liebler divergence of the product of their marginal distributions \( p(x)p(y) \), and the random variables’ joint distribution \( p(x,y) \):

\[
I(X,Y) = \int \int p(x,y) \log \frac{p(x,y)}{p(x)p(y)} \, dx \, dy,
\]

or, in a discrete case:

\[
I(X,Y) = \sum_{x} \sum_{y} p(x,y) \log \frac{p(x,y)}{p(x)p(y)},
\]

where \( X \) and \( Y \) represent, respectively, the set of all possible values for \( x \) and \( y \). If \( x \) and \( y \) are independent, \( p(x,y) = p(x) \ast p(y) \), and so IM is zero.

In its most basic form, and the one implemented in the current work, IM is a ranking algorithm, and its application results in the sorting of features according to their relevance when considering a certain label (AD, NC,...). Being \( F_i \) a given feature, with \( 1 \leq i \leq 309338 \), and \( Y \) the label vector containing the labels for all 896 subjects considered in this work, an IM vector was calculated by:

\[
IM(i) = I(F_i,Y), \quad \text{with} \quad 1 \leq i \leq 309338
\]

Sorting that vector leads to a ranking of the features according to their contribution to any given label.

D. Classification

This final step was based on the two semi-supervised algorithms presented in this work: ST and CT. These algorithms perform as wrapper functions, in the sense that they are based on the repetition of a supervised function, in this case, SVM.

1) Support Vector Machines

SVM are one of the most popular discriminative methods implemented in CAD systems. With its first developments in the thirties, the current form was only introduced in 1992 [17].

When considering a case of binary classification the SVM algorithm seeks the hyperplane that separates the data with maximum margin, or, the hyperplane that maximizes its distance to the closest training vectors of both classes (named support vectors). This separation is not always possible, and when so, SVM searches for the separation that minimizes classification errors. This type of SVM, called Soft Margin, was the one utilized in the current work, thanks to LIBSVM, a publicly available library for SVM made by Chang & Lin and available at [18].

2) Cross Validation

In order to successfully test the semi-supervised CAD systems suggested in this work, the existence of three distinct sets of data was essential: a test set, a training (labeled) set and the unlabeled set. With trying to produce impartial and unbiased results, the CAD systems should be judged by their performance on the classification of new data, different from the one used to train the classifiers. On the other hand, and since the suggested algorithms utilized SVM, there was a need to have a validation set with which to select the \( C \) parameter.

In a perfect situation the data available would be enough to suit all the needs. Since it wasn't, Cross Validation (CV) was implemented. \( K \)-fold CV randomly partitions datasets into \( k \) sub-sets of equal dimensions. Out of those, one is utilized as a test set while the other \( k-1 \) are set as useful data. This procedure is repeated \( k \) times with each of the \( k \) sub-sets used exactly once as the test set. Since, for the current work, an unlabeled dataset was also required, the solution adopted was to remove the L labeled elements needed from the useful data produced by each \( k \) fold, and defining the remaining data as the unlabeled set. CV was also performed in each iteration of the suggested algorithms to select the best \( C \) parameter of SVM in an impartial way.

3) Self-Training

The Self-training algorithm, also known as self-teaching or bootstrapping, probably due to its simplicity was one of first semi-supervised algorithms developed. With its first concepts appearing in the sixties, a more recent publication, in 1995, by
Yaroswky [19] is usually considered the foundation for the more recent developments. ST is a wrapper like algorithm in the sense that it is based on the repetition of supervised methods. Its performance is then dependent on the supervised method chosen. Being \( L \) a set of labeled data, and \( U \) a group of unlabeled elements, ST is defined by the following cycle:

1) Train a classifier \( g \) with the \( L \) training data;
2) Classify \( U \) elements with \( g \);
3) Find a sub-set of \( U', \) with the most confident classifications;
4) \( L = L + U'; \)
5) \( U = U - U'; \)
6) Repeat 1) until stopping criteria is met;

For the current work, SVM was the supervised method adapted to ST (function \( g \)). The confidence values were defined as the distance between the classified points and the separation hyperplane created by SVM (also true for Co-training). The main goal of the ST algorithm based on SVM is to, by the end of its cycle, arrive at an SVM classifier where its parameters \((w, \varepsilon \) and \( b)\) better "suit" the data it's trying to classify. The ST algorithm developed as part of the CAD system suggested in this work is presented in figure 4.

4) Co-Training

The Co-training algorithm is a semi-supervised learning technique introduced by Blum & Mitchell in 1998 [20]. Its basic idea, common to all semi-supervised methods, suggests the use of unlabeled data in order to improve the accuracy of classifiers that previously only used labeled data. Two attributes should be present in the data utilized to implement CT: there should be a good amount of unlabeled data available, and that data should have two different sets of information or "views" in which it can be perceived. As an example, lets consider the case where a webpage classifier can distinguish between sports and movie webpages, is demanded. To train such a system, manually labeled webpages would be required. However, that labeling is expensive, since it requires time and manual labor. Regarding those webpages, two different sets of information are available: the text within the pages themselves and the text associated with the hyperlinks that link to those pages. The CT algorithm suggests: create two weak classifiers based on the two types of information available about the data. For instance, the presence of the word "ball" in the webpage's text could be indicative of a sports webpage, while a hyperlink containing the word "critic" could potentially point to a movie webpage. By searching for webpages with "critic" in their hyperlink, their text could then be used to continue to train the classifier based on the webpage’s text and vice-versa.

Hereby, CT suggests the use of two classifiers and assumes that the features of the data can be split into two sub-sets that, on their own, are strong enough to train a classifier. To create these sub-sets, the main feature set \( X \) should be split into \( X_1 \) and \( X_2 \), in a way that \( X = X_1 \cup X_2 \) and \( X_1 \cap X_2 = 0 \). A given point \( x \) is then represented by \((x_1, x_2)\) in which \( x_1 \) and \( x_2 \) are vectors whose components represent the feature values of the sub-sets \( X_1 \) and \( X_2 \). Take \( L \) as a set of labeled training data made up of \( x \) points given by \((x_1, x_2)\), and \( U \) as a set of unlabeled data. The following cyclical process then defines the CT algorithm:

1) Use the data given in \( L \) to train a \( g_1 \) classifier that considers only the \( x_1 \) sub-set of \( x \);
2) Use the data given in \( L \) to train a \( g_2 \) classifier that considers only the \( x_2 \) sub-set of \( x \);
3) Let \( g_1 \) classify the elements of \( U \);
4) Let \( g_2 \) classify the elements of \( U \);
5) Select the \( D \) most confident classifications given by each of the classifiers \((D1 \) and \( D2)\);
6) Add \( D1 \) and \( D2 \) to \( L = L + D1 + D2 \);
7) Remove \( D1 \) and \( D2 \) from \( U = U - D1 - D2 \);
8) Repeat 1) until the stopping criteria is met;

For the present work, and given that the FDG-PET images utilized only present the \( V1 \) as features, the solution needed to then adapt CT to a CAD system was to split the available feature set. That partition was simply made by doubling the amount of features required for any given run of CT and then distributing, one by one, those features to the two classifiers developed for CT. The algorithm developed for the current work utilized SVM to create the \( g \) functions and is presented in figure 5.

---

**Figure 4 - Self-Training Algorithm**

<table>
<thead>
<tr>
<th>Self-Training</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Define:</strong></td>
</tr>
<tr>
<td>( L ) % Number of labeled subjects</td>
</tr>
<tr>
<td>( D ) % Number of subjects to be added in each iteration</td>
</tr>
<tr>
<td>( N, C ) %Vector indicating number and location of features</td>
</tr>
<tr>
<td>( \text{DATA:} ) given by the current k-fold of the CV process</td>
</tr>
<tr>
<td>- labeled data:</td>
</tr>
<tr>
<td>( X_{\text{labeled}} = {N, C \times L}; ) %feature vector of the ( L ) labeled subjects</td>
</tr>
<tr>
<td>( Y_{\text{labeled}} = {L}; ) %label vector for the ( L ) labeled subjects</td>
</tr>
<tr>
<td>- unlabeled data:</td>
</tr>
<tr>
<td>( X_{\text{unlabeled}} = {N, C \times U}; ) %feature vector of the ( U ) unlabeled subjects</td>
</tr>
<tr>
<td>( Y_{\text{unlabeled}} = {U}; ) %label vector for the ( U ) unlabeled subjects</td>
</tr>
<tr>
<td>- test data:</td>
</tr>
<tr>
<td>( X_{\text{test}} = {N, C \times T}; ) %feature vector of the ( T ) test subjects</td>
</tr>
<tr>
<td>( Y_{\text{test}} = {T}; ) %label vector for the ( T ) test subjects</td>
</tr>
<tr>
<td>- current data:</td>
</tr>
<tr>
<td>( X_{\text{current}} = X_{\text{labeled}} + \text{reserved space for} X_{\text{unlabeled}} )</td>
</tr>
<tr>
<td>( Y_{\text{current}} = Y_{\text{labeled}} + \text{reserved space for} Y_{\text{unlabeled}} )</td>
</tr>
<tr>
<td><strong>Algorithm:</strong></td>
</tr>
<tr>
<td>( C = {10, 10^2, \ldots, 10^n}; ) iteration = 0;</td>
</tr>
<tr>
<td>While ( \text{num_unlabeled} &gt; 0 ) do</td>
</tr>
<tr>
<td>CV(( k=10 )) of model made with SVM +C and with current data end</td>
</tr>
<tr>
<td>%best C value is determined</td>
</tr>
<tr>
<td>create SVM classifier model, with C and current data;</td>
</tr>
<tr>
<td>calculate ( \text{ACC}_{\text{test}} ) with model applied to test data(( N, C \times T ))</td>
</tr>
<tr>
<td>( \text{ACC}<em>{\text{test}}(\text{iteration}) = \text{acc}</em>{\text{model}}(\text{test data}); )</td>
</tr>
<tr>
<td>apply model to unlabeled data(( N, C \times U ));</td>
</tr>
<tr>
<td>rank classifications according to their confidence values;</td>
</tr>
<tr>
<td>( \text{current data} = \text{current data} + D ) most confident classifications. Confidence values also saved. ( D ) elements of each class will be added;</td>
</tr>
<tr>
<td>( \text{if confidence values of previous} D )’s &lt; present confidence values (of those ( D )’s) then</td>
</tr>
<tr>
<td>update those confidence values in current data and, if needed, correct label;</td>
</tr>
<tr>
<td>update values needed to calculate sensitivity, specificity and ( \text{ACC}_{\text{unlabeled}}; )</td>
</tr>
<tr>
<td>identify the ( D ) elements added, in unlabeled data as already &quot;added&quot; subjects;</td>
</tr>
</tbody>
</table>
| update \( \text{num_unlabeled, num_unlabeled=num_unlabeled - D; \)
The CAD systems based on ST and CT presented on chapter II were then applied to the FDG-PET database also presented in chapter II. In the current work, the systems developed were tested for three different cases:

- **ADvsNC**: In this case, AD and NC elements from the database were used as both labeled and unlabeled data and applied to the classification issue of trying to distinguish AD vs. NC patients;
- **MCI-CvsMCI-nC(MCI)**: For the second situation tested, the ST and CT algorithms were tested in a setting where MCI-C and MCI-nC were used to define both the labeled and the unlabeled sets, while trying to discriminate MCI-C vs. MCI-nC.

All the tests were performed with SVM based on the linear kernel, and the parameter C was set as \(10^{-8}, 10^{-6}, 10^{-4}, 10^{-2}, 10^{-1}, 2, 10, 20, 60, 80, 100, 200\). The tests were also performed for different sets of features. Out of the 309338 originally available features, and after the steps of feature extraction and selection, the number of features, N_C, was defined as N_C ∈ \{500, 2500, 5000, 7500, 12000, 24000, 48000\}, except for the MCI-CvsMCI-nC(AD&NC) case, where 48000 features weren't considered due to the increasingly bad results obtained with smaller feature sets.

As mentioned before, for both algorithms tested in this work, and for the three different cases considered, two parameters had to be defined: the size L of the labeled dataset and the number of D elements to be added in each iteration of both algorithms. Also requiring definition was the number of k-folds to be performed by the CV. To begin with, and considering the number of available elements from each label, the CV was defined as:

**ADvsNC**: Based on the fact that 584 elements were available in this case (292 ADs and 292 NCS), the number of k-folds was set to 6, resulting in approximately 98 test elements and 486 useful elements, in each fold;
- **MCI-CvsMCI-nC(AD&NC)**: Regarding the smaller size of available MCI elements, 312, k value was set to 3, resulting in approximately 104 and 208, test and unlabeled elements respectively, for each fold;
- **MCI-CvsMCI-nC(MCI)**: Following the same logic as in the previous case, the number of k-folds established for this case was also 3, even though this resulted in the case with the smallest number of iterations of ST and CT, since the 208 useful elements still had to be separated into labeled and unlabeled groups.

After setting the number of folds for each case, the number L was then set for each case:
- **ADvsNC**: Considering the 486 useful elements available, and trying to end up with a good number of unlabeled elements, L ∈ \{2, 10, 20, 60, 80, 100, 200\}, thus making the unlabeled dataset size vary between 484 and 286, for this case;
- **MCI-CvsMCI-nC(AD&NC)**: Since this case has a fixed number of unlabeled elements available (the 208 useful MCIs resultant of k-fold=3), and the L elements are selected from the 584 AD&NC subjects available in the database, L ∈ \{2,10,20,60,80,100,200\}, allowing for a sizeable set of AD&NC elements to be tested in the classification of MCI;
- **MCI-CvsMCI-nC(MCI)**: For the last case, and considering the need of separating the 208 useful MCIs in labeled and unlabeled datasets, L downsized to L ∈ \{2, 8, 12, 24, 30, 40, 60, 80\}.

Finally, and trying to allow for a good number of iterations to be performed in each case while also reducing computer simulation times without conditioning results, the number of elements D added in each iteration of both ST and CT was set as the unlabeled group. This setting was then applied to the classification of MCI-C vs. MCI-nC.

**MCI-CvsMCI-nC(MCI)**: For the last situation tested, the ST and CT algorithms were tested in a setting where MCI-C and MCI-nC were used to define both the labeled and the unlabeled sets, while trying to discriminate MCI-C vs. MCI-nC.

The number of subjects to be added in each iteration of ST and CT was set to \(10^2, 10^3, 10^4, 10^5\), respectively, for each fold; and the number of available elements from each label, the CV was defined as:

**ADvsNC**: Based on the fact that 584 elements were available in this case (292 ADs and 292 NCS), the number of k-folds was set to 6, resulting in approximately 98 test elements and 486 useful elements, in each fold;
as 4 for the ADvsNC case and 2 for both the MCI-CvsMCI-nC(AD&NC) and MCI-CvsMCI-nC(MCI) cases. All results presented here are the result of 24 repetitions, times the number of $k$-folds performed in each case.

A. Self-Training

During the tests performed with ST and with the goal of being able to evaluate the performance of the method presented, two kinds of accuracy were recorded throughout testing.

One of those accuracies, as mentioned before, was recorded while applying the SVM classifiers created in each iteration of ST, to the test dataset created in each fold by the CV. This accuracy was named ACC_test. For the first iteration of ST, the value of this accuracy, defined as ACC_test initial (ATI), is calculated while exclusive and solely utilizing the $L$ labeled dataset defined for that specific run, and so, it can be considered a supervised accuracy in the sense that it comes as a result of an application of a supervised system. The ATI value then serves as a baseline for the measure of ST’s performance, by allowing the ACC_test values obtained in the following iterations (that are a direct result of ST’s implementation) to be compared to that value. By the end of ST’s cycle, the last ACC_test value is then identified as ACC_test_final (ATF).

The other accuracy that is recorded follows the same logic applied to ACC_test but is applied to the unlabeled elements that are added during each iteration of ST. For the first iteration, this value is once again solely the result of the SVM classifier, created with the $L$ labeled elements, being applied to the unlabeled dataset, and is then identified as ACC_unlabeled_initial (AUI). During the rest of the ST cycle, the labels attributed to the $D$ elements added in each iteration, are recorded, to allow for the determination of the accuracy of all these classifications by the end of ST’s run. This accuracy is labeled as ACC_unlabeled_final (AUF). While determining AUF, the sensibility and specificity values associated with the classification of unlabeled elements are calculated.

1) ADvsNC

The first case tested for the ST algorithm was the ADvsNC one. First of all, and with trying to illustrate the set of performances ST produces, figure 6 is shown, exhibiting ACC_test’s evolution, for an $N_C$ of 12000.

By analyzing figure 6, one can conclude that, along with ST’s iterations, the value of ACC_test increases, and so, the addition of the unlabeled elements (now classified) is benefic to the CAD system.

For the present work, the type of performance presented in figure 6 was abridged to tables like table 2, where all the ATI and ATF values are registered for all possible combinations of $L$ and $N_C$.

![Figure 6 - Self-Training ADvsNC. ACC_test for 12000 features.](image)

Table 2 - ST ADvsNC, ACC_test values

Analyzing table 2 it is then possible to observe:

- When only 2 labeled elements were utilized, the ATI and consequently the ATF, values increase with the expansion of the $N_C$, reaching its maximum value for 12000 features, possibly due to the curse of dimensionality, since, for 24000 and 48000 there is a marked decrease of the ACC_test values. Nevertheless, it is worth noting that, with only 2 labeled subjects, the max ATF score reached was close to the max ATI verified when 20 labeled subjects were used, showing that ST presented gains similar to those brought by the inclusion of another 18 subjects;

- In most cases, for $L$ values higher than 2, the ATI value increases with the $N_C$ and then drops for 48000 features, except when 100 and 200 labeled elements were used where the higher number of available data supported a higher $N_C$. Again worth noting is the fact that, for an $L$ of 60, its max ATF score, 91.77%, is higher than the ATI registered for an $L$ of 100, 91.21%, again highlighting the ST algorithm;

- The highest values of ATF reached show that the gains introduced by ST go from 1.19% for an $L$ of 200, up to 9.17% for an $L$ of 2.

The values presented in table 3 show that ACC_unlabeled presented similar results than those shown by ACC_test. The sensibility and specificity values are too close to be able to say that the classification performance was higher either for AD or NC. The ST algorithm developed in this work presented gains of 1.1% for an $L$ of 200, up to 11.17% for an $L$ of 2. The highest value of AUF happened for the same $L$ and $N_C$ registered for the highest ATF, 200 and 24000, respectively.

![Table 3 - ST ADvsNC, AUI, AUF, Sens and Spec values.](image)

2) MCI-CvsMCI-nC(AD&NC)

In the second case where ST was tested, AD&NC labeled elements were tested in the classification of MCI-C vs. MCI-
nC (also the unlabeled set). By inspecting table 4 one can conclude that:

- For an L of 2 the max ATF was 77.78% while for an L of 8 that score was 79.43%. The ATI scores for these L values didn't follow any particular logic and that can probably be explained by the heterogeneity of the case at hand (classifying MCI subjects with few AD&NC labeled elements).

- With L values of 20 or higher the correlation of N_C and ACC_test was much more evident. The ATI and ATF values increased with the N_C, up to 12000 features, and then always dropped for an N_C of 24000. The max value of ATF was 83.55%, achieved for an L of 80. This value dropped for an L of 100 and climbed again for an L of 200 where it still wasn't as high as the one registered for 80. This shows that the number of labeled AD&NC subjects reaches a peak where it stops influencing the performance of MCI classification.

### Table 4 - ST MCI-CvsMCI-nC(AD&NC), ACC_test values

<table>
<thead>
<tr>
<th>L</th>
<th>ACC_test</th>
<th>L=500</th>
<th>L=1000</th>
<th>L=2000</th>
<th>L=3000</th>
<th>L=4000</th>
<th>L=6000</th>
<th>L=8000</th>
<th>L=10000</th>
<th>L=12000</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>67.04</td>
<td>77.78</td>
<td>75.67</td>
<td>76.07</td>
<td>76.14</td>
<td>78.48</td>
<td>77.48</td>
<td>79.89</td>
<td>79.00</td>
<td>77.62</td>
</tr>
</tbody>
</table>

By inspecting table 4 one can also conclude that the influence of the number of L labeled AD&NC elements in MCI classification was again noticeable in the fact that the AUF reached by an L of 60 was higher than any AUI registered. This time, the highest AUF wasn't registered in the same conditions of the max ATF.

### Table 5 - ST MCI-CvsMCI-nC(AD&NC), AUI, AUF, Sens and Spec values

<table>
<thead>
<tr>
<th>L</th>
<th>AUI</th>
<th>L=500</th>
<th>L=1000</th>
<th>L=2000</th>
<th>L=3000</th>
<th>L=4000</th>
<th>L=6000</th>
<th>L=8000</th>
<th>L=10000</th>
<th>L=12000</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>67.04</td>
<td>77.78</td>
<td>75.67</td>
<td>76.07</td>
<td>76.14</td>
<td>78.48</td>
<td>77.48</td>
<td>79.89</td>
<td>79.00</td>
<td>77.62</td>
</tr>
</tbody>
</table>

Once again, the ACC_unlabeled values followed a similar logic to ACC_test. The weak influence of the number of L labeled AD&NC elements in MCI classification was again noticeable in the fact that the AUF reached by an L of 60 was higher than any AUI registered. This time, the highest AUF didn't register in the same conditions of the max ATF.

B. Co-Training & Comparisons

During the tests performed with CT, the same ACC_test and ACC_unlabeled, sensibility and specificity recorded for ST were recorded, allowing for an easier comparison between the two methods.

1) ADvsNC

The first case where CT was applied was the ADvsNC one, where AD and NC elements were used as part of the unlabeled and the labeled, L, datasets. In figure 7, the ACC_test values obtained with the ST and CT algorithms are compared. Analyzing that figure, certain conclusions can be made:

- The results obtained with CT for an L value of 2 were superior to those achieved with ST, with max values of ATF of 87.41% and 85.98% respectively. This superiority vanished when the L value was 10 and 20, where the max ATF values presented by CT were 88.50% and 89.39% respectively, contrasting with 88.74% and 91.08%, values presented by ST; -When L was 60, 80 and 100, the CT algorithm achieved, once again, better results than ST, with maximum ATF scores of 91.97%, 93.02% and 93.24%, all achieved when the N_C was 24000. It is worth pointing out that, for CT, the ACC_test values produced with 48000 features were, for all values of L, worse than those achieved with 24000, mainly due to the nature of the algorithm that, by requiring a different set of features for both classifiers it uses, ends up having to resort to features that aren't that valuable; -With L set as 200, the ST method reached a max ATF of 95.52%, vs. 95.34% presented by CT, both for an N_C of 24000, even though, in the same conditions, CT produced superior values of AUF, sensibility and specificity (table 8).
Evaluating table 8, for low $L$ values, CT produced maximum ACC_unlabeled scores for an $N_C$ of 7500 or lower, while with $L$ values from 60 to 200 the top scores were gotten for an $N_C$ of 24000, coinciding with the conditions seen for the top ACC_test values.

Comparing the ACC_unlabeled results obtained for both CT, table 8, and ST, table 3, is it possible to notice that CT reached higher values of AUF for all values of $L$ except 20, with the same being applied to Sens and Spec (table 9).

2) MCI-CvsMCI-nC(AD&NC)

By direct comparison of the scores presented by both methods, and for the classification of MCI-C vs MCI-nC, with unlabeled data using AD and NC elements, CT produced better ACC_test results for all $L$ values tested, except when $L$ was 10:

- With an $L$ of 2, CT outscored ST with a max ATF of 79.35% vs. the 77.78% produced by ST, both for an $N_C$ of 500;
- For $L$ from 20 to 200, while ST achieved max values of ATF for an $N_C$ of 24000, CT was much more irregular, with max values achieved at different $N_C$;
- For an $L$ of 20, CT increased the max ATF score of ST by 0.22%, while for an $L$ of 60 the increase was of 0.32% (from 82.85% up to 83.17%);
- Once again exposing the frail influence of the number of labeled AD&NC elements in the classification of MCIs, the biggest ATF was achieved for an $L$ of 80, and an $N_C$ of 12000, with a score of 84.14%, corresponding to an increase from ST of 0.59%.

<table>
<thead>
<tr>
<th>Number of Labeled Elements</th>
<th>L=2</th>
<th>L=10</th>
<th>L=20</th>
<th>L=50</th>
<th>L=100</th>
<th>L=200</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spec</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sens</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 9 - CT vs ST, ADvsNC. Best AUF, Sens and Spec scores.

![Figure 7 - ACC_test, CT vs ST ADvsNC](image)
Regarding table 10, the largest AUF score was, in this case, not achieved for the same conditions of the best ATF, with max AUF occurring when the \( L \) was 200 and \( N_C \) 5000.

Comparing ACC_unlabeled values of the ST algorithm, table 5, to those reached by CT, table 10, it is noticeable that, as was the case for ACC_test, CT achieved better scores for all \( L \) values except 10, with a maximum of 84.14% corresponding to a Sens. of 84.32% and a Spec of 84.11%.

Comparing the results shown in the previous table with the ones achieved by ST, table 7, it can then be said that CT produced better results for all \( L \) values except 12 and 30.

In the MCI-CvsMCI-nC(MCI) case, and as is exposed by table 12, the biggest AUF score developed for the same conditions as the biggest ATF, \( L \) of 80 and \( N_C \) being 24000.

Comparing the results shown in the previous table with the ones achieved by ST, table 7, it can then be said that CT produced better results for all \( L \) values except 12 and 30.
IV. CONCLUSIONS & FUTURE WORK

With the current work, two semi-supervised classification methods, self-training and co-training, were tested in CAD systems with the goal of AD classification. These two methods were tested in their ability to distinguish between AD vs. NC, and MCI-C vs. MCI-nC.

Self-training achieved, in the ADvsNC case, margin gains of up to 11.17% in classification accuracy, when compared to classifiers trained with only labeled elements. The final accuracy achieved with only 2 labeled elements was comparable to those found in the literature for supervised methods. In the MCI-CvsMCIC-nC, a case of extreme importance but a more demanding one and less researched, MCI-CvsMCI-nC(AD&NC) clearly produced worse results than MCI-CvsMCI-nC(MCI), with maximum accuracy values of 83.55% and 90.01% respectively, corresponding to an ST gain of 2.91% and 2.16% (again when compared to classifiers trained only with the labeled subjects).

The good ST results were, for the most part, improved by Co-training that, for the ADvsNC case presented score improvements for Sens. and Spec. of 0.24% and 0.18%, hitting maximum values of 95.69% and 96.01%. In the MCI-CvsMCI-nC(AD&NC) case, CT improved the max ATF by 0.59% when comparing the 84.14% value achieved to the 83.55% accuracy presented by ST. The MCI-CvsMCI-nC(MCI) case saw smaller ACC improvements, with maximum Sens. and Spec. values of 90.96% and 90.61% being hit by CT.

Future work could expand on the ideas presented in this work by implementing new ways to try and minimize the shortcomings found in this work. The implementation of boundary levels in the SVM classifier should produce, at the expense of ignoring some elements, better values of ACC.

Also, adapting the CT algorithm to use two different biomarkers, like MRI and FDG-PET, could provide interesting results. Finally, adapting the CAD system to a totally unsupervised algorithm could end up producing good results, by eliminating the subjectivity of the classification given by specialists.

REFERENCES


