Automatic detection of Mild Cognitive Impairment using high-dimensional acoustic features in spontaneous speech

Cong Zhang[1], *Wenxing Guo*[2], *Hongsheng Dai*[1]

[1]Newcastle University, UK
[2]Essex University, UK

cong.zhang@newcastle.ac.uk, wg22745@essex.ac.uk, hongsheng.dai@newcastle.ac.uk

Abstract

This study addresses the TAUKADIAL challenge, focusing on the classification of speech from people with Mild Cognitive Impairment (MCI) and neurotypical controls. We conducted three experiments comparing five machine-learning methods: Random Forests, Sparse Logistic Regression, k-Nearest Neighbors, Sparse Support Vector Machine, and Decision Tree, utilizing 1076 acoustic features automatically extracted using openSMILE. In Experiment 1, the entire dataset was used to train a language-agnostic model. Experiment 2 introduced a language detection step, leading to separate model training for each language. Experiment 3 further enhanced the language-agnostic model from Experiment 1, with a specific focus on evaluating the robustness of the models using out-of-sample test data. Across all three experiments, results consistently favored models capable of handling high-dimensional data, such as Random Forest and Sparse Logistic Regression, in classifying speech from MCI and controls.

Index Terms: TAUKADIAL challenge, dementia, mild cognitive impairment, cognitive assessment

1. Introduction

1.1. Mild cognitive Impairment

With ageing emerging as an increasingly prevalent global trend, cognitive impairment has become an ever more concerning topic. The scarcity of medical resources poses significant challenges for the timely detection of such cognitive impairment as dementia. The early stage of dementia is commonly identified as mild cognitive impairment (MCI). MCI often, though not necessarily, progresses into full-blown dementia. People with MCI may self-report subjective cognitive difficulties, e.g. forgetting recent events, yet their speech typically does not exhibit obvious distinctions from peers of a similar age. Given the neurodegenerative nature of these conditions, early intervention is crucial. However, the subtle nature of early-stage symptoms makes accurate diagnosis challenging. Therefore, automatic assessment emerges as a valuable tool, offering easy accessibility for people with MCI and a convenient reference for clinicians.

The review in [1] presents thoroughly the distinction between MCI and typical speech, summarising the effects of semantics, syntactic complexity, fluency, vocal parameters, and pragmatics. For speech-related factors, MCI speakers exhibit less fluent speech patterns, different intonation for complex descriptions, and different speech rates and phonation patterns.

1.2. Automatic classification

Compared with the classification of dementia (e.g. [2, 3]), relatively early-stage MCI has received less attention. [4] trained classification models for MCI and Alzheimer's disease with acoustic features, including Automatic Speech Recognition outputs (i.e. text transcriptions with filled and unfilled pauses) and other linguistic features such as morphological or semantic features. The combination of both acoustic and other linguistic features outperformed either unimodal input. They also compared acoustic features extracted based on manual and automatic segmentation and found that manual segmentation had higher accuracy. In [5], temporal features (such as duration of utterance, number and length of pauses) and spectral features (F0, F1, and F2) were used to build logistic regression models to predict MCI, cognitive impairment, MCI with global cognitive impairment and the controls. The MCI classification accuracy, assessed through a 3-fold cross-validation, yielded a result of 0.61. More recently, as a part of the TAUKADIAL Challenge, [6] tested `eGeMAPs` features (88 features), a self-supervised feature learning approach, `wav2vec` (512 features), and the two combined. After a 20-fold cross-validation, the highest performing model, `wav2vec + eGeMAPs`, presented a UAR of 59.18.

1.3. The current study

This paper is submitted as a part of the TAUKADIAL Challenge, focusing on creating models to classify speech from people with MCI and controls. To achieve this goal, we set out to investigate: (1) what acoustic features are needed to successfully predict the classification; (2) what modelling technique can best classify the data; (3) whether a fully automated process, i.e. without any manual annotation and labelling, can lead to an effective classification.

2. Methods

2.1. Data

The data used in this study was distributed as 'training data' by the TAUKADIAL organizers[1]. The full dataset is described in [6]. This subset of data contain 387 files from 129 distinct speakers (three files per speaker), including English data (American English) and Chinese data (Taiwan Mandarin). The language variety information was not given explicitly.

2.2. Language identfication

The Python package `VoxLingua107` [7] was used for language identification. The English data, being clearer and louder, resulted in mostly accurate detection; the Mandarin data included speakers speaking with various accents, and therefore the detection results were not ideal. Consequently, we recorded

[1]Test set details have not been published thus not included.

the identification results for English and classified the rest as Mandarin. Out of 383 files (129 speakers), 182 files (62 speakers) were identified as English, while the remaining 201 files (67 speakers) were categorized as Mandarin.

2.3. Feature extraction

We extracted acoustic features from the raw audio files employing the openSMILE Toolkit [8], which was specifically developed for audio feature extraction and classification of speech and music signals. Two feature sets, `emobase` (988 features) and `eGeMAPSv02` (88 features), were extracted using the Python package `opensmile`[2] in Python 3.9. These two feature sets provide sufficient coverage of consonant and vowel features, voice quality features, and prosodic features, which are all relevant to MCI speech. We did not process these data any further to keep the process as automated as possible.

2.4. Feature selection

We employed regularization methods for feature selection, which is a popular machine-learning technique aimed at reducing overfitting and improving the generalization ability of models by adjusting the weights of features. This typically involves introducing a regularization term into the loss function, which is a function of the feature weights, to constrain the magnitude of feature weights or induce sparsity. See more details in the description of the SLR and SSVM methods in Section 2.5.

2.5. Classification methods

Considering the high-dimensional nature of the dataset, where the number of features exceeds the number of samples and significant correlations exist among the data, we employed five model training methods: Random Forests (RF), Sparse Logistic Regression (SLR), k-Nearest Neighbors (KNN), Sparse Support Vector Machine (SSVM), and Decision Tree (DT).

Random Forests is a classifier comprised of a collection of tree-structured classifiers [9]. It constructs a multitude of decision trees on randomly selected sub-samples of the original dataset and aggregates the results from each tree to make predictions. During the construction of each tree, a random subset of features is considered for node splitting to increase model diversity. The prediction process involves employing a voting mechanism for classification tasks. For classification, the predicted class $\hat{y}$ is determined by:

$$\hat{y} = \arg\max_k \sum_{i=1}^{n} I(\hat{y}_i = k),$$

where $\hat{y}_i$ represents the predicted class by the i th tree, n is the total number of trees, k is the class label, and $I(\cdot)$ is the indicator function.

Sparse Logistic Regression is a method that combines Elastic Net regularization with logistic regression [10]. The method uses both L_1 regularization (Lasso) and L_2 regularization (Ridge) to overcome some limitations of each individual method. In logistic regression, the objective function of Elastic Net is typically defined as:

$$-\frac{1}{n}\sum_{i=1}^{n}[y_i \log(p_i) + (1-y_i)\log(1-p_i)] + \lambda_1||\beta||_1 + \lambda_2||\beta||_2^2,$$

where n is the number of samples, y_i is the observed response (class labels), p_i is the predicted probability for each observation, λ_1 and λ_2 are hyperparameters controlling the strength of L_1 and L_2 regularization penalties, $||\beta||_1$ is the L_1 norm (sum of absolute values), $||\beta||_2^2$ is the squared L_2 norm (sum of squared parameters), and β is the coefficient vector of the model. By tuning the values of λ_1 and λ_2, one can balance the model's fit and complexity, leading to improved generalization performance. The main advantage of Elastic Net is its ability to handle high-dimensional data and correlated features, while selectively retaining some relevant features and compressing others to zero.

Sparse Support Vector Machine is an extension of the traditional Support Vector Machine algorithm aimed at inducing sparsity in the solution [11]. The objective function of SSVM is typically defined as:

$$\frac{1}{n}\sum_{i=1}^{n}\text{hingeLoss}\left(y_i\left(x_i'w+b\right)\right) + \text{penalty}(w),$$

where x_i represents the feature vector of the i th sample, y_i denotes the true label of the i th sample, w is the weight vector of the model, b stands for the bias term of the model and hingeLoss $(t) = \max(0, 1-t)$ is the hinge loss function. For this method, we still applied the Elastic Net penalty. Namely, penalty$(w) = \lambda_1||w||_1 + \lambda_2||w||_2^2$. SSVM aims to find a hyperplane that separates the classes while using only a subset of the features, effectively reducing the dimensionality of the problem and improving computational efficiency.

k-Nearest Neighbors classification method is a non-parametric and instance-based learning algorithm used for classification tasks. It operates by classifying an unseen data point based on the majority class among its k nearest neighbors in the feature space. The choice of k, the number of neighbors to consider, is a crucial hyperparameter that influences the model's performance. Larger values of k result in smoother decision boundaries but may lead to less flexibility, while smaller values of k provide more flexible decision boundaries but may lead to overfitting.

The Decision Tree algorithm is a popular machine learning method used for both classification and regression tasks. It builds a tree-like structure by recursively splitting the dataset into subsets based on the feature that provides the most significant information gain or decrease in impurity. Each internal node of the tree represents a decision based on a feature, while each leaf node corresponds to a class label or a regression value. Decision trees are interpretable and easy to visualize, making them useful for understanding the decision-making process.

2.6. Evaluation metrics

We used the following metrics to evaluate the MCI classification task. Specificity (σ): $\sigma = \frac{T_N}{T_N+F_P}$; Precision ($\pi$): $\pi = \frac{T_P}{T_P+F_P}$; F_1 score: $F_1 = \frac{2\pi\rho}{\pi+\rho}$; Balanced accuracy (unweighted average recall, UAR): $\text{UAR} = \frac{\sigma+\rho}{2}$, where $\rho = \frac{T_P}{T_P+F_N}$ denotes sensitivity that measures the proportion of actual positive cases, N is the number of patients, T_P represents the number of true positives, T_N denotes the number of true negatives, F_P is the number of false positives and F_N stands for the number of false negatives. These metrics collectively provide a comprehensive assessment of the performance of MCI classification.

[2] https://github.com/audeering/opensmile-python

3. Experiments

In Experiment 1, the models were trained with all data, including English and Chinese, and tested on an in-sample test set. Experiment 2 included a language detection step and the models were trained separately. Experiment 3 further enhanced the language-agnostic model in Experiment 1, but tested the robustness of the models using out-of-sample test data. All models were trained and tested in R [12]. Some relevant R packages were used to compare these methods: Random Forests (R package `ranger`[13] with parameters: num.trees = 500, mtry=the square root of the number of variables), Sparse Logistic Regression (R package `glmnet`[14] with parameters: alpha=0.5, family = "binomial", s = 0.001, type = "response"), k-Nearest Neighbors (R package `class` [15] with parameter: k = 5), Sparse Support Vector Machine (R package `sparseSVM` [16] with parameter: lambda = 0.01), Decision Tree (R package `rpart` [17] with parameter: method = "class"). Data and scripts are available[3].

3.1. Experiment 1

In Experiment 1, all available data were used for model training. A test set of 100 samples (out of 387) were randomly selected from the same training dataset for evaluating the trained models.

The results in Table 1 demonstrate that RF and SLR have exceptional performance with perfect precision, specificity, F1 score and UAR of 100%, followed by DT with high precision, specificity, F1 score, and UAR of 98%, 97%, 94%, and 94% respectively. SSVM exhibits moderate performance with precision, specificity, F1 score, and UAR of 84%, 75%, 90%, and 86% respectively, while KNN shows comparatively lower performance across all metrics.

Table 1: *Comparison of different methods used for training data from English and Chinese speech*

Method	Precision	Specificity	F1 score	UAR
RF	100%	100%	100%	100%
DT	98%	97%	94%	94%
KNN	77%	68%	78%	73%
SLR	100%	100%	100%	100%
SSVM	84%	75%	90%	86%

100 samples are randomly selected from the same training dataset for evaluating the trained models.

3.2. Experiment 2

In Experiment 2, we conducted separate analyses for English and Chinese data using the predicted language labels. Test sets of 100 samples (out of 201) from the Chinese data and 100 samples (out of 186) were randomly selected from the same training data for model evaluation.

The results are shown in Tables 2 and 3. From these two tables, we can observe that RF, SLR and SSVM perform the best among the methods considered, while KNN consistently exhibits the poorest performance among all the methods used. The results from these language-specific models showed consistently high accuracy in the classification task as the language-agnostic models, which suggests that (1) language-agnostic models are powerful enough to make the classifications; and (2) smaller training data size did not influence the robustness of the models, especially the best-performing ones.

[3] Anonymous link for peer review: https://tinyurl.com/n3nkepzr

Table 2: *Comparison of different methods used for training data from Chinese speech*

Method	Precision	Specificity	F1 score	UAR
RF	100%	100%	100%	100%
DT	87%	86%	92%	91%
KNN	77%	82%	68%	71%
SLR	100%	100%	100%	100%
SSVM	99%	99%	98%	98%

100 samples are randomly selected from the same training dataset for evaluating the trained models.

Table 3: *Comparison of different methods used for training data from English speech*

Method	Precision	Specificity	F1 score	UAR
RF	100%	100%	100%	100%
DT	93%	87%	92%	89%
KNN	75%	41%	82%	66%
SLR	100%	100%	100%	100%
SSVM	100%	100%	100%	100%

100 samples are randomly selected from the same training dataset for evaluating the trained models.

3.3. Experiment 3

In Experiment 3, we further evaluated the predictive performance of these methods. The entire dataset was partitioned into training and test sets, using train-test ratios of 1:1 (i.e. 1 part allocated for training and 1 part for testing), 2:1, 3:1, 4:1 and 5:1 respectively. To enhance the stability of the models, the original dataset was randomly partitioned 100 times for each train-test split, and we conducted 100 rounds of training and testing for each method using random cross-validation.

Figures 1, 2 and 3 present the results of this experiment. From these three line graphs, we can see that RF consistently outperforms the other methods, exhibiting the highest scores across all three metrics. Specifically, in terms of F1 Score (Fig. 1), RF achieves scores ranging from 0.84 to 0.88, followed by DT with scores ranging from 0.73 to 0.78; and then SLR and SSVM with scores ranging from 0.78 to 0.81. KNN consistently shows lower F1 scores compared to other methods, ranging from 0.66 to 0.67.

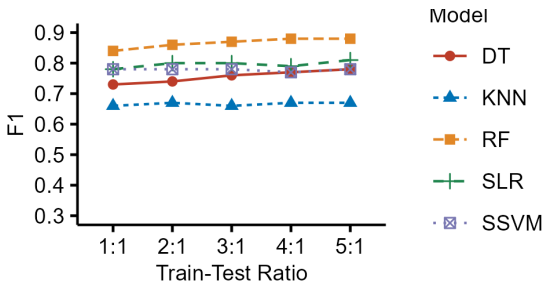

Figure 1: *F1 based on data split at random for 100 times. Train-test ratio denotes the proportions of the training and test sets.*