AUTOMATIC SELECTION OF SPOKEN LANGUAGE BIOMARKERS FOR DEMENTIA DETECTION

KE Xiaoquan

PhD

The Hong Kong Polytechnic University

2024

The Hong Kong Polytechnic University

Department of Electrical and Electronic Engineering

Automatic Selection of Spoken Language Biomarkers for Dementia Detection

KE Xiaoquan

A thesis submitted in partial fulfilment of the requirements for the degree of Doctor of Philosophy

August 2023

CERTIFICATE OF ORIGINALITY

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written, nor material that has been accepted for the award of any other degree or diploma, except where due acknowledgment has been made in the text.

__________________________________(Signed)

___________KE Xiaoquan___________(Name of student)

Abstract

Automatic Selection of Spoken Language Biomarkers for Dementia Detection

Dementia is a severe cognitive impairment that may affect older adults' health and daily lives and burden their families and caretakers. The most common form of dementia is Alzheimer's disease (AD). Currently, dementia can be diagnosed through brain imaging, identification of apolipoprotein E genotypes, measuring the level of brain-derived neurotrophic factors, cerebrospinal fluid exams, and other laboratory measures. However, these measures are invasive and costly. As dementia also manifests itself as spoken language deficits, effective detection of early signs of the disease through the analyses of spoken languages can facilitate timely intervention to slow deterioration. This thesis analyzes a diverse set of features extracted from spoken languages and selects the most discriminative ones for dementia detection. We refer to these features as spoken language biomarkers of dementia.

This thesis proposes two deep-learning-based feature ranking (FR) methods, called dual dropout ranking (DDR) and dual-net feature ranking (DFR), to rank and select features. DDR and DFR are based on a dual-net architecture that performs feature selection (FS) and dementia detection by two neural networks: Operator and Selector. The two networks are alternatively and cooperatively trained to optimize the performance of both FS and dementia detection. Specifically, in DDR, the operator is trained on features obtained from the selector to reduce classification loss, and the selector is optimized to predict the operator's performance based on automatic regularization. DDR ranks features according to the probabilities that the corresponding features should be purged (or kept). In DFR, the selector is trained to find multiple subsets of features to predict the operator's performance, and the operator uses

these feature subsets to minimize classification errors. DFR uses all of the selector's parameters to determine the contributions of individual features to the selector's predictions, taking into account the non-linear relationship between the input variables and the network's output. It allows for evaluating the contributions of individual input variables in a multi-layer neural network with non-linear activation functions. We also proposed a two-step FS approach that utilizes filter methods to pre-screen features and applies more expensive FS methods to rank the pre-screened features.

The proposed FR methods were evaluated on three dementia datasets – ADReSS, AD2021, and JCCOCC-MoCA. Results on ADReSS and AD2021 show that the full feature set comprises many redundant features and that feature ranking can improve the accuracy of dementia detection. In particular, using the most discriminative features discovered by DDR, we achieved an F_1 score of 90.4% on the ADReSS test set, which surpasses the official baseline performance by 15.9 percentage points. Similarly, using the most discriminative features discovered by DDR, we achieve an F_1 score of 86.7% on the AD2021 test set, surpassing the official baseline performance by 8.1 percentage points. The evaluations on the JCCOCC-MoCA dataset show that DFR can significantly reduce feature dimensionality while identifying small feature subsets with performance comparable or superior to the whole feature set. The selected features have been uploaded to https://github.com/kexquan/AD-detection-Feature-selection, and codes are aviable at https://github.com/kexquan/dual-dropout-ranking and https://github.com/kexquan/dual-net-feature-ranking.

Author's publications:

1. Xiaoquan Ke, Man-Wai Mak, Jinchao Li, and Helen M. Meng, "Dual Dropout Ranking of Linguistic Features for Alzheimer's Disease Recognition", in Proceeding of Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA), Tokyo, Dec. 2021, pp. 743–749.

2. Xiaoquan Ke, Man-Wai Mak, Helen M. Meng, "Automatic Selection of Discriminative Features for Dementia Detection in Cantonese-Speaking People", in Proceeding of Interspeech, Incheon, Sep. 2022, pp. 2153–2157.

3. Xiaoquan Ke, Man-Wai Mak, and Helen M. Meng, "Feature Selection and Text Embedding For Detecting Dementia from Spontaneous Cantonese", in Proceeding of 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Rhodes Island, Jun. 2023.

4. Helen Meng, Brian Mak, Man-Wai Mak, Helene Fung, Xianmin Gong, Timothy Kwok, Xunying Liu, Vincent C. T. Mok, Patrick Wong, Jean Woo, Xixin Wu, Ka Ho Wong, Shensheng Xu, Naijun Zheng, Ranzo Huang, Jiawen Kang, Xiaoquan Ke, Junan Li, Jinchao Li, Yi Wang, "Integrated and Enhanced Pipeline System to Support Spoken Language Analytics for Screening Neurocognitive Disorders", in Proceedings of Interspeech, Dublin, Aug. 2023.

5. Xiaoquan Ke, Man-Wai Mak, and Helen M. Meng, "Jointly Modelling Transcriptions and Phonemes with Optimal Features to Detect Dementia from Spontaneous Cantonese", in Proceeding of Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA), Taipei, 2023.

6. Xiaoquan Ke, Man-Wai Mak, and Helen M. Meng, “Automatic Selection of Spoken Language Biomarkers for Dementia Detection”, *Neural Networks*, vol. 129, Jan. 2024, pp. 191–204.

7. Sean Shensheng Xu, Xiaoquan Ke, Man-Wai Mak, Helen M. Meng, and Timothy C.Y. Kwok, “Speaker-turn Aware Diarization for Speech-Based Cognitive Assessments”, *Frontiers in Neuroscience*, Jan. 2024.

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my supervisor Dr. M.W. MAK, who processes expertise in speech processing, machine learning, deep learning, and bioinformatics. I deeply admire his considerable patience and great earnestness on my academic research. Throughout the process of my academic publications, Dr. M.W. MAK has been an invaluable mentor, offering invaluable guidance and constructive criticism. His feedback on draft manuscripts has been instrumental in refining my writing style and enhancing the clarity and coherence of my ideas. His expertise, willingness, and insightful feedback have also significantly shaped the quality and direction of this thesis. I would like extend my gratitude to Dr. Helen M. Meng, for her unwavering support, guidance, and invaluable mentorship throughout this research journey.

I would like to thank Dr. Sean Shensheng Xu, Dr. Wei-Wei Lin, Dr. You-Zhi Tu, Lu Yi, and Jin-Cao Li for their collaborative spirit and camaraderie. Their willingness to engage in intellectual discussions and offer alternative perspectives enriched my understanding of the research. I would like to extend my appreciation to the university administration, the department of Electronic and Information Engineering, and libraries for their extensive resources and services that have facilitated access to the literatures and materials necessary for this research. Lastly, but certainly not the least, I am eternally grateful to my family and friends for their unwavering support and belief in my abilities.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS/ABBREVIATIONS

ACC accuracy

AD Alzheimer's disease

ADR active data representation

ADReSS AD Recognition Through Spontaneous Speech Challenge

AFS active feature selection

ASR automatic speech recognition

balanced-ACC balanced accuracy

BiLSTM bidirectional long-short term memory

CNNs convolutional neural networks

CoCoGen complexity contour generator

CRI comprehensive relative importance

CV cross-validation

DDR dual dropout ranking

DFR dual-net feature ranking

DFS deep feature selection

DropoutFR dropout feature ranking

DT decision trees

eGeMAPS extended Geneva Minimalistic Acoustic Parameter Set

F0 fundamental frequency

FDR Fisher's discriminant ratio

FIR feature importance ranking

FR feature ranking

FS feature selection

HC healthy control

HCs healthy controls

HNR harmonic to noise ratio

k-nn k-nearest neighbor

LIWC linguistic inquiry and word count

LLDs low-level descriptors

LSP line spectral pairs

MAE mean absolute error

major NCD major neurocognitive disorders

MCI mild cognitive impairment

MFB multimodal factorized bi-linear (pooling)

MFCCs mel-frequency cepstral coefficients

MFH multimodal factorized high-order (pooling)

mild NCD mild neurocognitive disorders

MoCA Montreal Cognitive Assessment

mRMR minimal-redundancy-maximal-relevance

MutInfo mutual information

NAS neural architecture search

NLP natural language processing

OOV out-of-vocabulary

PCA principal component analysis

PD Parkinson's disease

PeaCorr Pearson's correlation

POS part-of-speech

PRE precision

REC recall

RF random forest

RMS root-mean-square

SBS sequential backward selection

SFS sequential forward selection

SVM support vector machines

SVM-RFE SVM recursive feature elimination

TR training partitions/training partition

TS test partitions/test partition

VAD voice activity detector

ViT vision Transformer

WPM words-per-minute

LIST OF SYMBOLS

Filter methods

X, Y**:** random variable

$\mathcal{S}$**:** feature set

a, b**:** random variable

r_{ab}**:** Pearson correlation (PeaCorr) coefficient of a and b

$\bar{a}$**:** the mean of the variable a

$\bar{b}$**:** the mean of the variable b

n**:** the number of samples (observations)

μ_j^+**:** the mean of the j-th feature belonging to the positive class

μ_j^-**:** the mean of the j-th feature belonging to the negative class

σ_j^+**:** the standard derivation of the j-th feature belonging to the positive class

σ_j^-**:** the standard derivation of the j-th feature belonging to the negative class

Deep Feature Selection (DFS)

ψ**:** parameters of the deep neural network

$\boldsymbol{x}$**:** the input variable vectors

$\boldsymbol{w}$**:** weight coefficent of the one-to-one layer

$\boldsymbol{W}^{(k)}$**:** weight matrix of the k-th layer

λ_1, λ_2**:** mixing parameters between the $L1$ and $L2$ penalties

α_1, α_2**:** mixing parameters for regularization that reduce model complexity

Dropout Feature Ranking (DropoutFR)

θ**:** dropout rate

$\boldsymbol{\theta}$**:** dropout rate vector

z**:** dropout mask

$\boldsymbol{z}$**:** dropout mask vector

$\boldsymbol{x}$**:** input feature vector

d**:** dimension of the input feature vector

$\boldsymbol{y}$**:** target

$\mathcal{M}$**:** a mini-batch with $|\mathcal{M}|$ pairs of $\boldsymbol{x}_i$ and $\boldsymbol{y}_i$

$\odot$**:** element-wise multiplication

λ**:** regularization coefficient

$\boldsymbol{u}$**:** $\boldsymbol{u} \in \mathbb{R}^d$ follows the Uniform$(\mathbf{0}, \mathbf{1})$ distribution

t**:** a normalization constant, which is set to 0.1

Feature Importance Ranking (FIR)

$\boldsymbol{x}$**:** input feature vector

$\boldsymbol{y}$**:** target

$\mathcal{M}$**:** a mini-batch with $|\mathcal{M}|$ pairs of $\boldsymbol{x}_i$ and $\boldsymbol{y}_i$

$\odot$**:** element-wise multiplication

$\boldsymbol{z}$**:** feature mask vector

$\mathcal{Z}$**:** feature mask subset comprising $|\mathcal{Z}|$ of $\boldsymbol{z}$

ψ**:** the operator's parameters

φ**:** the selector's parameters

Dual Dropout Ranking (DDR)

θ**:** dropout rate

$\boldsymbol{\theta}$**:** dropout rate vector

z**:** dropout mask

$\boldsymbol{z}$**:** dropout mask vector

$\mathcal{Z}$**:** dropout mask subset that comprising $|\mathcal{Z}|$ of $\boldsymbol{z}$

$\boldsymbol{x}$**:** input feature vector

d**:** dimension of the input feature vector

$\boldsymbol{y}$**:** target

$\mathcal{M}$**:** a mini-batch with $|\mathcal{M}|$ pairs of $\boldsymbol{x}_i$ and $\boldsymbol{y}_i$

ψ**:** the operator's parameters

φ**:** the selector's parameters

$\odot$**:** element-wise multiplication

$\boldsymbol{u}$**:** $\boldsymbol{u} \in \mathbb{R}^d$ follows the Uniform$(\mathbf{0}, \mathbf{1})$ distribution

t**:** a normalization constant, which is set to 0.1

Dual-net Feature Ranking (DFR)

$\boldsymbol{x}$**:** predictor variabels/input feature vector

d**:** dimension of the predictor variabels/input feature vector

$\hat{\beta}_j, j \in \{1, 2, \cdots, d\}$**:** parameters of the linear regression model

$\hat{\beta}_0$**:** bias of the linear regression model

y**:** output of the 1-layer fully-connected network

$\boldsymbol{w}$**:** weight vector of the 1-layer fully-connected network

$\boldsymbol{c}$**:** feature importance vector

l_1**:** number of nodes of the hidden layer

$\boldsymbol{W}^{(1)}$**:** weight matrix of the hidden layer

$\boldsymbol{b}^{(1)}$**:** bias vector of the hidden layer

$\boldsymbol{o}^{(1)}$**:** output of the hidden layer

$\boldsymbol{w}^{(2)}$**:** weight vector of the output layer

$\boldsymbol{W}^{(i)}, i = 1, 2, \ldots, L-1$**:** weight matrix for the i-th hidden layer

$\boldsymbol{w}^{(L)}$**:** weight vector for the output layer

$\boldsymbol{z}$**:** feature mask vector

$\boldsymbol{z}_1$**:** the optimal feature mask vector

$\boldsymbol{z}_2$**:** the best feature mask vector

$\mathcal{Z}$**:** feature mask subset

$\boldsymbol{z}_j, j \in \{3, \cdots, |\mathcal{Z}|\}$**:** candidate feature mask vectors

$\boldsymbol{y}$**:** target

$\mathcal{M}$**:** a mini-batch with $|\mathcal{M}|$ pairs of $\boldsymbol{x}_i$ and $\boldsymbol{y}_i$

ψ**:** the operator's parameters

φ**:** the selector's parameters

$\odot$**:** element-wise multiplication

Chapter 1

INTRODUCTION

Dementia is a severe cognitive impairment that may seriously affect the health and daily lives of the afflicted individuals. The greatest known risk factor for dementia is increasing age, and the most common form of dementia is Alzheimer's disease (AD). According to a report from the World Health Organization,[1] more than 55 million people live with dementia worldwide, and there are nearly 10 million new cases every year. In 2019, the estimated global societal cost of dementia was $1.3 trillion, and these costs are expected to surpass $2.8 trillion by 2030. It was reported that 33% of seniors died with AD or dementia. It is the 6th leading cause of death. In the USA, 10% of Americans over the age of 65 were diagnosed with AD, and 66% among the diagnoses are women. The disease has a huge impact on the quality of life, not only for individuals with dementia but also for their families and caretakers. Fortunately, with effective detection of early dementia, disease-modifying medications and interventions are possible [5]. Early detection of dementia can facilitate intervention to slow the disease progression.

Currently, dementia can be diagnosed through brain imaging [6], identification of apolipoprotein E genotypes [7], measuring the level of brain-derived neurotrophic factors [8], cerebrospinal fluid exams [9], and other laboratory measures. Studies have found that dementia-induced language impairment could be found in patients years before the disease was diagnosed [10]. Research also showed that individuals

[1] https://www.who.int/news-room/fact-sheets/detail/dementia

with progressive cognitive decline exhibit subtle linguistic impairment even in the presymptomatic stages of the disease [11]. These findings suggest that individuals with dementia display language deficits in the preclinical stages of the disease, indicating that such deficits may manifest even before the clinical diagnosis is made. Consequently, early detection of dementia can be achieved through speech and language analyses. Following early detection, interventions and disease-modifying medications can be implemented. Although the disease progression cannot be reversed, patients may experience a decelerated disease progression, enhanced quality of lives, decreased medical expenses, and extended lifespans.

1.1 Automatic Detection of Dementia through Speech and Language Analyses

Recently, automatic detection of dementia through speech and language analyses has gathered attention in the research community. Dementia detection involves feature extraction followed by classification. The features can be generally grouped into speech-based and transcription-based, depending on the source of extraction. Speech-based features are extracted from speech recordings. They characterize atypical changes in the speaker's voicing, such as decreasing pitch, decreasing jitter, shorter speech segments, etc. Transcription-based features, on the other hand, are extracted from transcriptions. They can be divided into linguistic, semantic, and pragmatic features, indicating language impairments or language deficits in the patients' spoken language.

1.1.1 Speech-Based Features

Speech-based features, such as voice quality [12,13], verbal reaction time, and silence duration [14], are typically extracted from speech recordings. These features contain a variety of acoustic characteristics of the speakers.

Haider *et al.* [15] compared different types of paralinguistic acoustic features, including eGeMAPS [16], ComParE 2013 [17], Emobase [17], and MRCG [18] for dementia detection. As the paralinguistic acoustic features are high-dimensional, Pearson's correlation (PeaCorr) tests were performed to reduce the feature dimensions. They performed dementia detection at the segment-level. More specifically, the full audio recording of a subject was split into several short speech segments, and different paralinguistic acoustic features were extracted from these segments. A method called active data representation (ADR) was adopted to model the acoustic information of the full audio recording using the features of all speech segments. Majority voting was applied to the predicted labels of different speech segments to make the final decisions.

Nasreen *et al.* [19] distinguished AD patients from non-AD control of similar age using speech-based features. They used two types of speech-based features: interactional features and acoustic features. The former characterizes the temporal and interactional aspects of conversations, which include pauses less than 1.5 seconds, pauses greater than 1.5 seconds, gaps, lapses, attributable silences, etc. The acoustic features includes pitch, amplitude, energy, and mel-frequency cepstral coefficients (MFCCs). They achieved a promising accuracy of 87% using the interactional features alone.

In addition to eGeMAPS features [16], Gauder *et al.* [20] investigated different speech-based embeddings for automatic detection of AD. The speech-based embeddings are high-level representations extracted from pre-trained models, such as the trill model [21], the Allosaurus model [22], and the Wav2Vec 2.0 model [23]. The authors also performed segment-level classification and obtained the final score for an audio file by averaging the scores over all speech segments. The pre-trained Wav2Vec 2.0 model was also adopted by Balagopalan *et al.* [24] for recognizing English-speaking AD patients. They fed speech segments to the pre-trained Wav2Vec 2.0 model and extracted the embeddings from the model. They obtained embeddings for an audio file by averaging the embeddings over all speech segments.

Generally speaking, speech-based features used for dementia detection include paralinguistic acoustic features (eGeMAPS, ComParE, Emobase, MRCG, etc.), conventional hand-crafted acoustic features (pitch, amplitude, energy, MFCCs, interactional features, etc.), and speech-based embeddings extracted from pre-trained models (trill, Allosaurus, and Wav2Vec 2.0). The paralinguistic acoustic features are adopted from other speech-related tasks, such as speaker recognition and emotion recognition. The conventional hand-crafted acoustic features are specially designed for dementia detection. The speech-based embeddings are high-level representations extracted from pre-trained models designed for high-level tasks. For example, the Allosaurus model was designed for phone recognition and Wav2Vec 2.0 is an end-to-end speech-to-text model. The extraction of speech-based embeddings avoids the manual design of features.

1.1.2 Transcription-Based Features

In addition to the speech-based features, diverse transcription-based features have also been used for dementia detection. The transcription-based features are extracted from either the automatic or manual transcriptions, which capture the semantic, syntactic, and lexical aspects of the speaker's utterances.

Qiao *et al.* [25] combined linguistic complexity and disfluency features with Transformer-based pre-trained language models for AD detection tasks. The disfluency features (silent pauses, speed of articulation, filled pauses, and pronunciation) containing the speakers' articulatory characteristics were extracted from automatic speech recognition (ASR) systems. The linguistic complexity features (syntactic complexity, lexical richness, register-based n-gram frequencies, and information-theoretic measures) were obtained by analyzing the transcriptions using a complexity contour generator (CoCoGen). The pre-trained BERT [26] and ERNIE [27] models were also fine-tuned using the transcriptions for dementia detection.

The Transformer-based language models were extensively investigated by Syed *et* *al.* [28] for dementia detection. They compared the efficacy of BERT and its derivatives, including DistilBERT [29] and RoBERTa [30] for capturing the structural and linguistic properties of the transcriptions. The BART [31] model was also included. Instead of taking the entire transcription as a single entity, the authors in [28] generated token-level embeddings and computed the embeddings for the entire transcription by applying statistics pooling on the token-level embeddings. They introduced a special pre-processing step that integrates silence durations into the transcriptions. Specifically, when the duration was between 2s and 4s, they added `<uhm>` to the transcriptions. If the silence was between 4s and 6s, they added `<uhm uhm>`. If the silence exceeded 6s, they added `<long silence>`. They also combined handcrafted features – including syntactic, readability, and lexical diversity – with the embeddings for recognizing AD patients.

Yuan *et al.* [32] applied a special pre-processing step that encodes pauses in the transcriptions for AD detection. More precisely, the pauses were divided into three groups according to their durations: G_1 (pauses less than 0.5s), G_2 (pauses between 0.5s and 2s), and G_3 (pauses longer than 2s). Three groups of pauses were encoded using three punctuations `<.>`, `<..>`, and `<...>`, respectively. Finally, the BERT and ERNIE models were fine-tuned using the pre-processed transcriptions as input.

In a recent study, Li *et al.* [33] explored the capabilities of large Transformer models for AD detection. Rather than directly fine-tuning a model to differentiate between healthy older adults and AD patients, the authors proposed perturbing a small GPT-2 model [34] to create an artificially degraded GPT model called GPT-D. Then, AD patients were detected by computing the perplexity ratio between the two models, given the spoken languages of an unknown subject. The idea is based on the notion that the perturbation of GPT-2 induces linguistic anomalies related to dementia, causing the model to generate text with characteristics associated with AD.

Generally speaking, recent studies utilized pre-trained language models (BERT, ERNIE, DistilBERT, RoBERTa, BART, etc) to automatically capture the structural and linguistic properties of transcriptions. These studies usually take special pre-processing steps to incorporate dementia-related information (self-repair terms, edit terms, short pauses, long pauses, etc.) into the transcriptions. The hand-crafted linguistic, lexical, and syntactic features were also used for dementia detection because of their ability in revealing the patients' abnormal language characteristics.

1.1.3 Combined Speech-Based and Transcription-Based Features

Some studies [35] combined speech-based and transcription-based features, e.g., paralinguistic features were combined with linguistic fluency features. Another approach is to fuse the decisions from multiple modalities. For instance, in [36], the modalities include acoustic, cognitive, and linguistic, and in [37, 38], the modalities comprise acoustic and textual domains.

Fraser *et al.* [39] used a large number (370) of features to capture different linguistic phenomena. The features include part-of-speech (POS) statistics, syntactic complexity, grammatical constituents, psycholinguistics-related (familiarity, imageability, and age-of-acquisition), vocabulary richness, information content, and repetitiveness. Additionally, several acoustic features indicative of pathological speech and a set of features based on MFCCs were extracted. To identify the fundamental structure in the data, the authors applied factor analysis, which decomposed 50 highly correlated features into four factors: semantic impairment, acoustic abnormality, syntactic impairment, and information impairment.

Rohanian *et al.* [40] used bidirectional long-short term memory (BiLSTM) to model speech-based and transcription-based features. They used the COVAREP feature set [41] as the speech-based features. The transcription-based features were text embeddings extracted from a pre-trained GloVe model [42]. They also took a special pre-processing step that integrates disfluency information (self-repairs and edit terms) and pause information (short and long pauses) into the transcriptions.

Chatzianastasis *et al.* [43] employed a multimodal approach to combining convolutional neural networks (CNNs) and pre-trained language models. The authors used the approach to process speech recordings (as the acoustic modality) and their corresponding transcriptions (as the text modality). The speech recordings were converted to images with three channels (log-Mel spectrograms, delta, and delta-delta), which were then inputted to the CNNs. To address the challenge of designing an efficient CNN architecture, they leveraged neural architecture search (NAS) for identifying high-performing CNN architectures. The authors also explored various fusion methods, such as multimodal factorized bi-linear (MFB) pooling [44] and multimodal factorized high-order (MFH) pooling [44], to combine the speech and text modalities.

Recently, there has been increased interest in combining the acoustics, transcription, and speech modalities. For example, to distinguish patients with progressive neurodegenerative memory disorders from those with non-progressive functional memory disorders, Mirheidari *et al.* [45] extracted 12 speech features and 12 transcription features. More importantly, they also incorporated 20 conversational features specifically designed for characterizing the differences between speaker turns. In a different study, Ilias *et al.* [46] used Transformer models to combine the transcription and speech modalities. For the transcription modality, they employed a BERT model to automatically capture the semantic, syntactic, and lexical aspects of the speaker's transcriptions. For the speech modality, they converted the speech signals into log-Mel spectrograms and utilized a vision Transformer (ViT) [47] to automatically extract features from the spectrograms. Zhu *et al.* [48] explored transfer learning across multiple domains, including image, audio, speech, and language, to distinguish patients with AD from healthy controls (HCs). They employed various transfer learning models, such as MobileNet [49] (image), YAMNet [50] (audio), Mockingjay [51] (speech), and BERT (language). MFCCs were used as inputs for the MobileNet, and log-Mel spectrograms were used as the input to the Mockingjay model.

1.2 Relevance of Various Features for Dementia Detection

Several studies have investigated the relevance of various features for dementia detection. The general process is feature extraction followed by feature selection (FS). For example, Ammar *et al.* [52] extracted transcription-based features to identify the early onset of AD. The researchers classified these features into three distinct categories: syntactic, semantic, and pragmatic. Syntactic features are derived from the usage of nouns, pronouns, adjectives, and verbs. Semantic features include type-token ratio and idea density. Pragmatic features include degree of paraphrasing, number of repetitions, and the number of syllables spoken per minute. The authors compared the performance of three types of classifiers: support vector machines (SVM), decision trees (DT), and neural networks. They selected discriminative features from the transcription-based features using three FS methods, including information gain, k-nearest neighbor (k-nn), and SVM recursive feature elimination (SVM-RFE). FS was performed on the entire training data to compute the ranking weights for all features and delete the features with small weight. Experimental results reveal that all classifiers exhibited improved performance using these FS methods.

Weiner *et al.* [53] extracted a range of speech-based and transcription-based features from biographic interviews for AD state screening. Speech-based features comprise pause-based attributes, speaking rate characteristics, and i-vectors [54]. Transcription-based features consist of lexical richness, linguistic inquiry and word counts (LIWC), POS tags, and perplexities. To reduce the dimensionality, a nested forward FS method was employed. Firstly, a leave-one-subject-out cross-validation (CV) was used to generate the training partitions (TR) and test partitions (TS) of the folds. Secondly, on the TR of each fold, forward FS was conducted using a second-level leave-one-subject-out CV. The best-performing set of features was determined from each forward FS. Finally, the selection frequency of each feature was obtained by counting how many times the feature was selected in the nested CV. Their experiments show that transcription-based features were often selected and among them LIWC, conversational POS, i-vectors, and written POS are the most often. Additionally, they also applied nested forward FS to select features in biographic interviews to predict AD development in five years [55].

Alhanai *et al.* [56] extracted demographic, speech-based, and transcription-based features to identify discriminative features for cognitive impairment detection. The demographic features included the subject's age, sex, highest level of self-reported education, and occupation. The openSMILE toolkit [17] was used for extracting the acoustic features, which include information on the subject's pitch, probability of voicing, root-mean-square (RMS) energy, MFCCs, harmonic to noise ratio (HNR), zero crossing rate, shimmer, jitter, and the difference between the features in neighboring frames. The transcription-based features comprise the number and duration of words per speaking turn, words-per-minute (WPM), the number of question marks (`<?>`) and hesitation marks (`<um>`) per speaking turn, the number of unique words, and the out-of-vocabulary (OOV) ratio. A two-step FS method was adopted in a nested leave-one-subject-out CV to select features. Firstly, a leave-one-subject-out CV was utilized to divide the data into TR and TS partitions. Secondly, on the TR of each fold, PeaCorr was applied to pre-screen the original feature sets, which was followed by a binomial logistic-regression model regularized by an elastic-net [57] to select discriminative features. Finally, feature importance was determined using the coefficients of the regularized logistic regression model, and the selection frequencies of individual features were also reported. The experimental results show that cognitive impairment is positively correlated with decreasing pitch, decreasing jitter, short speech segment, and an increasing number of question marks (`<?>`).

Table 1.1 summarizes the studies that perform FS for dementia detection. Haider *et al.* [15] used speech-based features only, whereas Ammar *et al.* [52] employed transcription-based features only. Haider *et al.* [15] and Ammar *et al.* [52] performed FS using the entire training data prior to CV, whereas Weiner *et al.* [53, 55] and

Alhanai *et al.* [56] performed FS in a nested leave-one-subject-out CV.

Table 1.1: Summary of the studies that investigate the relevance of various features for dementia detection.

	Extracted Feature	**FS Method**	**FS Procedure**	**Selected Feature**
Haider *et al.* [15]	eGeMAPS, ComParE 2013, Emobase, and MRCG	PeaCorr	Selecting features on the entire training data	Features that were not correlated with the duration of the speech chunks
Ammar *et al.* [52]	Syntactic, semantic, and pragmatic	Information gain, kNN, and SVM-RFE	Selecting features on the entire training data	The use of nouns, the use of pronoun, repetition, etc.
Weiner *et al.* [53]	Pause-based, speaking rate, i-vectors, lexical richness, LIWC, POS tags, and perplexity	Forward FS	Nested leave-one-subject-out CV	LIWC, POS, and i-vectors
Weiner *et al.* [55]	Pause-based, speaking rate, i-vectors, lexical richness, POS tags, LIWC, perplexity, between-speaker perplexity	Forward FS	Nested leave-one-subject-out CV	LIWC, POS, perplexity, between-speaker perplexity, i-vectors, and pause-based
Alhanai *et al.* [56]	Demographic, speech-based, and transcription-based features	PeaCorr, and elastic-net	Nested leave-one-subject-out CV	Decreasing pitch, decreasing jitter, shorter speech segment, and an increasing number of question marks (<?>)

Recent studies also identified biomarkers for healthcare and bioinformatics applications. Shen *et al.* [58] selected important genes from high-dimensional gene expression data. They studied the limitations of the $L1$ penalization-based methods such as Lasso [59] and elastic-net [57] in identifying highly correlated features. To address the limitations, they proposed comprehensive relative importance (CRI) [58] analysis independent of the sample size and matrix rank. The CRI was proven to be more effective in selecting relevant features in some high-dimensional biological datasets compared to the $L1$ penalization-based methods. Qin *et al.* [60] assessed the speech and language impairments in Cantonese-speaking individuals with aphasia. They proposed some text features and combined them with conventional acoustic features to encompass various aspects of the impairments. A two-tailed paired t-test was utilized to select ASR-aligned text features. Additionally, they used Spearman's correlation tests to select acoustic features that are highly corrected with the target. Palmerini *et al.* [61] aimed to identify the patients with early mild Parkinson's disease (PD) by characterizing their postural behavior. A total of 175 features were extracted from the accelerometer signals to quantify tremor, acceleration, and displacement of body sway. They performed FS by exhaustively searching the subsets of these features. Their results indicate that several subsets of these features achieve misclassification rates as low as 5%, showcasing the efficiency of FS in improving the accuracy of PD diagnosis. Their results also emphasize the potential of their method for monitoring the progression of PD.

1.3 Research Objectives

As previously proposed, FS is utilized to determine the feature relevance for dementia detection. This study also utilized FS to detect dementia. We refer to the selected features as spoken language biomarkers of dementia. Detecting dementia through FS offers several advantages. 1) FS helps in reducing the feature dimensions, which may lead to a more accurate model. 2) FS can improve explanations. By selecting the most relevant features, researchers and clinicians can understand which features are indicative of dementia. 3) FS can offer insights into disease mechanisms. By focusing on the selected features, we can have a deeper understanding of cognitive changes in dementia patients. 4) FS can improve detection performance.

We summarize our research objectives as follows.

1. Diverse spoken language features can contribute to dementia detection, and our study focuses on identifying the most discriminative features for detection.

2. Combining different features can cause difficulties in determining the feature relevance. To mitigate these difficulties, we explore FS methods to select the spoken language biomarkers and better determine the feature relevance.

3. To streamline a feasible and general process to select the spoken language biomarkers, we investigate *automatic* FS methods.

4. To improve detection performance, we develop more advanced FS methods, especially the deep-learning-based methods.

Chapter 2

FEATURE SELECTION

We are now in the era of big data, with a vast amount of ubiquitous, high-dimensional data in various fields, such as social media, healthcare, and bioinformatics. When data-mining and machine-learning algorithms are applied to high-dimensional data, the high feature-dimensionality will easily cause overfitting in machine learning models, making the models unable to generalize to unseen data. In bioinformatics, the high feature-dimensionality also casts difficulty in interpreting the input variables. Feature selection (FS), a powerful dimension reduction technique, can address overfitting and interpretability issues. Unlike principal component analysis (PCA), which reduces feature dimensionality by projecting feature vectors onto a low-dimensional space, FS reduces feature dimensionality by selecting a feature subset from the original feature set. Therefore, the selected features maintain physical meanings and provide interpretability.

This chapter investigates various FS methods, especially the deep-learning-based methods.

2.1 Conventional Feature Selection Methods

2.1.1 Filter Methods

Filter methods are usually computationally less expensive and do not require training. The filter methods are also independent of the machine learning models for classification. In the cases where the feature dimension is very high, filter methods are indispensable because they can produce a reduced set of features that can be further selected by other expensive FS methods.

The minimal-redundancy-maximal-relevance (mRMR) criterion [62] selects features by assessing the max-relevance and min-redundancy between the features and the targets. Given two random variables X and Y, their mutual information (MutInfo) is defined as:

$$I(X;Y) = \iint p(x,y) \log \frac{p(x,y)}{p(x)p(y)} \mathrm{d}x\mathrm{d}y, \tag{2.1}$$

where $p(x)$ is the probability density function of X, and $p(x,y)$ is the joint density of X and Y. Max-relevance is to find a feature set that maximizes the mean of the MutInfo between individual features x_i and the class c:

$$\max_{\mathcal{S}} D(\mathcal{S},c), \text{ where } D(\mathcal{S},c) = \frac{1}{|\mathcal{S}|} \sum_{x_i \in \mathcal{S}} I\left(x_i; c\right). \tag{2.2}$$

Min-redundancy is to select a mutually-exclusive feature set that minimizes the dependence between the features:

$$\min_{\mathcal{S}} R(\mathcal{S}), \text{ where } R(\mathcal{S}) = \frac{1}{|\mathcal{S}|^2} \sum_{x_j, x_k \in \mathcal{S}} I\left(x_j, x_k\right). \tag{2.3}$$

Combining max-relevance and min-redundancy, the mRMR criterion is:

$$\max_{\mathcal{S}} \Phi(D,R), \text{ where } \Phi = D - R. \tag{2.4}$$

We can employ MutInfo for FS. When applying the MutInfo filter method, we calculate the MutInfo between each feature and the target (class) variable. Afterwards, we rank the features based on their respective MutInfo.

The Pearson correlation (PeaCorr) tests measure the linear relationships between two random variables a and b. The PeaCorr coefficient (r_{ab}) can be calculated using the formula:

$$r_{ab} = \frac{\sum_{j=1}^{n}(a_j - \bar{a})(b_j - \bar{b})}{\sqrt{\sum_{j=1}^{n}(a_j - \bar{a})^2 \sum_{j=1}^{n}(b_j - \bar{b})^2}}, \tag{2.5}$$

where $\bar{a}$ and $\bar{b}$ represent the mean of the variable a and variable b respectively for n observations. When using the PeaCorr as a filter method for FS, we find the redundant features by sorting the PeaCorr coefficients of feature pairs.

The Fisher's discriminant ratio (FDR) [63, 64] selects features by assessing the within-class means and variances of each candidate feature. The formula for FDR is:

$$\text{FDR}(j) = \frac{(\mu_j^+ - \mu_j^-)^2}{(\sigma_j^+)^2 + (\sigma_j^-)^2}, \tag{2.6}$$

where μ_j^+, μ_j^-, σ_j^+, and σ_j^- represent the class-conditional means and standard derivations of the j-th feature, respectively. Features with high $\text{FDR}(j)$ are selected.

2.1.2 Wrapper Methods

The wrapper methods assess the relevance of features according to their learning performance on a classifier or a regression model. For example, sequential forward selection (SFS) iteratively finds the best features that lead to maximum performance gain. On the other hand, sequential backward selection (SBS) iteratively removes features that do not have a significant effect on the performance. SVM-RFE [65] ranks the coefficients of a linear SVM to eliminate features.

2.1.3 Embedded Methods

The embedded methods use the intrinsic structure of a learning algorithm to embed FS into the underlying model. For example, the Lasso [59] imposes $L1$ penalty on the coefficients of a regression model [1]. Group Lasso [66, 67] make certain groups of feature weights to be close to zero, effectively selecting or disregarding specific groups of features altogether. Sparse Group Lasso [68] selects several group of features, but not all the features in the selected group will be kept. Figure 2.1 illustrates the differences between Lasso, Group Lasso, and Spare Group Lasso.

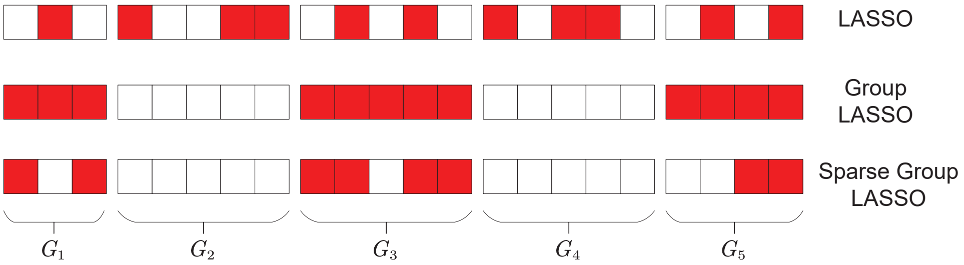


Figure 2.1: The differences between Lasso, Group Lasso, and Spare Group Lasso. Lasso encourages sparsity at individual-feature level. Group Lasso selects several groups of features, i.e., G_1, G_3, and G_5. Sparse Group Lasso also selects G_1, G_3, and G_5, but some features in these two groups are discarded. Adopted from [1].

The elastic-net regularization [57] places $L1$ and $L2$ penalties on the coefficients of a regression model to encourage sparsity. Random forest (RF) [69] determines feature importance by evaluating the extent to which each feature reduces impurity.

2.2 Deep-Learning-Based Methods

2.2.1 Deep Feature Selection (DFS)

Sparsity regularization can also be adopted in deep learning models for FS. For example, in deep feature selection (DFS) [2], elastic-net regularization [57] is imposed on the coefficients between the input and the first hidden layer. The architecture of DFS is depicted in Figure 2.2.

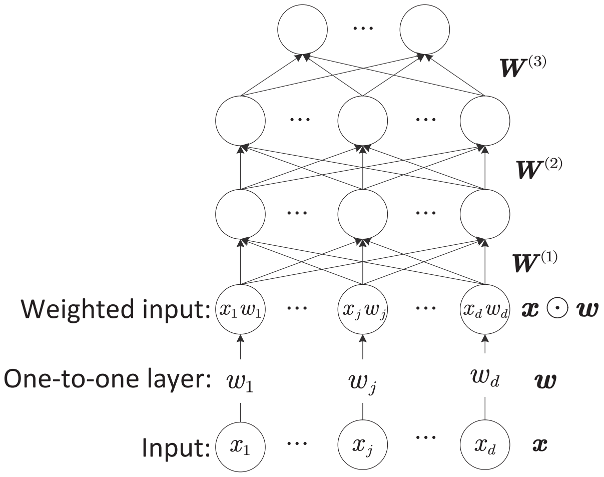

$W^{(3)}$
$W^{(2)}$
$W^{(1)}$
Weighted input: x_1w_1 ⋯ x_jw_j ⋯ x_dw_d $x \odot w$
One-to-one layer: w_1 w_j w_d w
Input: x_1 ⋯ x_j ⋯ x_d x

Figure 2.2: The architecture of DFS. There is a one-to-one correspondence between the coefficients in $\boldsymbol{w}$ and the input variables in $\boldsymbol{x}$. $\boldsymbol{w}$ is a sparse vector. Adopted from [2].

The learning objective of DFS is:

$$\mathcal{L}(\boldsymbol{x};\psi) = l(\boldsymbol{x};\psi) + \lambda_1\left(\frac{1-\lambda_2}{2}\|\boldsymbol{w}\|_2^2 + \lambda_2\|\boldsymbol{w}\|_1\right) + \alpha_1\left(\frac{1-\alpha_2}{2}\sum_{k=1}^{K+1}\left\|\boldsymbol{W}^{(k)}\right\|_F^2 + \alpha_2\sum_{k=1}^{K+1}\left\|\boldsymbol{W}^{(k)}\right\|_1\right), \tag{2.7}$$

where $\psi = \{\boldsymbol{w}, \boldsymbol{W}^{(1)}, \ldots, \boldsymbol{W}^{(K+1)}\}$ contains the network parameters, $l(\boldsymbol{x};\psi)$ is the cross-entropy loss for classification, $\boldsymbol{w}$ contains the coefficients corresponding to the input variables in $\boldsymbol{x}$, $\lambda_1\left(\frac{1-\lambda_2}{2}\|\boldsymbol{w}\|_2^2 + \lambda_2\|\boldsymbol{w}\|_1\right)$ is the elastic-net regularization (the combination of $L1$ and $L2$ penalties), and λ_1 and λ_2 are mixing parameters between the $L1$ and $L2$ penalties. The second regularization $\alpha_1\left(\frac{1-\alpha_2}{2}\sum_{k=1}^{K+1}\left\|\boldsymbol{W}^{(k)}\right\|_F^2 + \alpha_2\sum_{k=1}^{K+1}\left\|\boldsymbol{W}^{(k)}\right\|_1\right)$ is another elastic-net-like term that helps reduce the model complexity and speed up optimization. There is a one-to-one correspondence between the coefficients in $\boldsymbol{w}$ and the input variables in $\boldsymbol{x}$. Therefore, the coefficients $\boldsymbol{w}$ can reflect the importance of the corresponding input variables.

2.2.2 Dropout Feature Ranking (DropoutFR)

There are other deep-learning-based methods using regularization, such as dropout feature ranking (DropoutFR) [4]. These methods rank the features according to the probabilities (dropout rates) that the features should be purged (or kept). The higher the dropout rates, the lower the rank of the features. Given a dropout rate vector $\boldsymbol{\theta} = (\theta_1, \theta_2, \ldots, \theta_j, \ldots, \theta_d)$ and a dropout mask vector $\boldsymbol{z} = (z_1, z_2, \ldots, z_j, \ldots, z_d)$, we denote the distribution of $\boldsymbol{z}$ as

$$q(\boldsymbol{z}) = \prod_{j=1}^{d} q\left(z_j \mid \theta_j\right) = \prod_{j=1}^{d} \operatorname{Bern}\left(z_j \mid \theta_j\right), \tag{2.8}$$

where θ_j is the dropout rate for the j^{th} feature, and $z_j \in \{0, 1\}$ is the corresponding binary dropout mask. This gives us a fully factorized Bernoulli distribution for feature-wise feature ranking (FR). Before training, the dropout rate corresponding to each input feature is set to 0.5. During the training process, the dropout rates can be optimized to determine the feature ranks. After training, each feature will have a different dropout rate, and the feature rank is based on the dropout rate. For instance, the feature with a dropout rate of 0.1 ranks higher than the feature with a dropout rate of 0.9, because the latter is more likely to be purged during the training process. The idea of using dropout rates to rank features is novel because it considers the rank of the features in a probabilistic measure instead of a deterministic measure. Different from deterministic measures that use some coefficients to determine the feature rank, DropoutFR uses probabilities to indicate the FR.

The learning objective of DropoutFR is to minimize the following loss:

$$\mathcal{L}(\mathcal{M}; \boldsymbol{\theta}) = -\frac{1}{|\mathcal{M}|} \sum_{i=1}^{|\mathcal{M}|} l\left(\boldsymbol{x}_i \odot \boldsymbol{z}_i, \boldsymbol{y}_i\right) + \frac{\lambda}{|\mathcal{M}|} \sum_{i=1}^{|\mathcal{M}|} \sum_{j=1}^{d} z_{ij}, \tag{2.9}$$

where $\boldsymbol{\theta}$ contains trainable dropout rates that determine $\boldsymbol{z}_i$ probabilistically, $\mathcal{M}$ is a mini-batch with $|\mathcal{M}|$ pairs of feature vector $\boldsymbol{x}_i$ and target $\boldsymbol{y}_i$, $\boldsymbol{z}_i$ is the dropout mask corresponding to $\boldsymbol{x}_i$, $\odot$ is element-wise multiplication, and $l(\cdot)$ is the standard cross-entropy loss for classification. The second term is a regularization (penalty) term that encourages sparsity on the dropout masks: λ is a regularization coefficient and d is the dimension of the input vector. The regularization on the dropout masks is similar to LASSO regularization because both of them encourage sparsity on the coefficients. We can see that most of the dropout masks will become 0 (sparse) if the regularization works well. The regularization term minimizes the number of preserved features by encouraging sparsity on the dropout masks.

A key point is that DropoutFR optimizes the dropout rates $\boldsymbol{\theta}$ through the concrete relaxation:

$$\boldsymbol{z}(\boldsymbol{\theta}) = \text{sigmoid}\left(\frac{1}{t}\left[\log\boldsymbol{\theta} - \log(\mathbf{1}-\boldsymbol{\theta}) + \log\boldsymbol{u} - \log(\mathbf{1}-\boldsymbol{u})\right]\right), \tag{2.10}$$

where $\boldsymbol{u} \in \mathbb{R}^d$ follows the Uniform$(\mathbf{0}, \mathbf{1})$ distribution and t is a normalization constant, which is set to 0.1 in our experiments. Eq. 2.10 relaxes the discrete dropout mask to a continuous function of the dropout rates, which enables the optimization of the dropout rates through back-propagation.

2.2.3 Feature Importance Ranking (FIR)

There are other deep-learning-based methods that utilize different strategies for regularization. For example, the feature importance ranking (FIR) [3] is based on a dual-net architecture that combines FS and classification via two neural networks (selector and operator). The dual-net architecture of FIR is depicted in Figure 2.3.

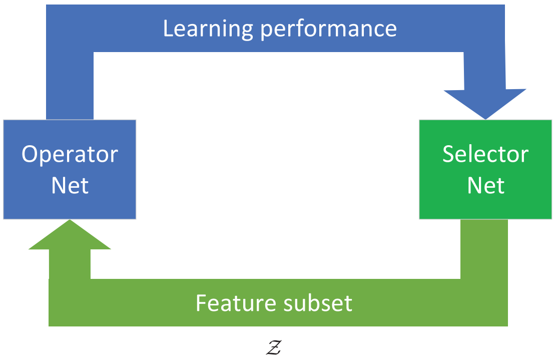


Figure 2.3: The dual-net architecture of FIR. Adopted from [3].

Figure 2.3 shows that FIR contains two neural networks to simultaneously incorporate FS and classification. The learning objective of FIR is:

$$\begin{aligned}
\mathcal{L}_O\left(\mathcal{Z},\mathcal{M};\psi\right) &= \frac{1}{|\mathcal{M}|\,|\mathcal{Z}|}\sum_{\boldsymbol{z}\in\mathcal{Z}}\sum_{(\boldsymbol{x},\boldsymbol{y})\in\mathcal{M}} l(\boldsymbol{x}\odot\boldsymbol{z},\boldsymbol{y};\psi) \\
\mathcal{L}_S\left(\mathcal{Z};\varphi\right) &= \frac{1}{2\,|\mathcal{Z}|}\sum_{\boldsymbol{z}\in\mathcal{Z}}\left(f_S(\boldsymbol{z};\varphi)-\frac{1}{|\mathcal{M}|}\sum_{(\boldsymbol{x},\boldsymbol{y})\in\mathcal{M}} l(\boldsymbol{x}\odot\boldsymbol{z},\boldsymbol{y};\psi)\right)^2,
\end{aligned}
\tag{2.11}$$

where $\mathcal{L}_O\left(\mathcal{Z},\mathcal{M};\psi\right)$ is the operator's learning objective, ψ contains the operator's parameters, $\mathcal{M}$ is a mini-batch with $|\mathcal{M}|$ pairs of feature vector $\boldsymbol{x}_i$ and target $\boldsymbol{y}_i$, and $\mathcal{Z}$ is the feature mask subset with size $|\mathcal{Z}|$. $\mathcal{L}_S\left(\mathcal{Z};\varphi\right)$ is the selector's learning objective, φ contains the operator's parameters, and $f_S(\boldsymbol{z};\varphi)$ is the output of the selector. The operator is trained to minimize the classification loss based on the features obtained from the selector. In each iteration of training, the operator first obtains the feature mask subset $\mathcal{Z}$ from the selector, and the selected features are obtained from $\boldsymbol{x}\odot\boldsymbol{z}$. The operator's learning performance based on the selected features is obtained and passed to the selector as a feedback indicating how well the operator performs on the features selected by the selector. The selector is trained to select the optimal feature set for predicting the operator's learning performance. The selector uses input gradient [70] to rank the features and selects the optimal feature set. The regularization for FS is achieved by selecting a subset of features from the original feature set based on the input gradient.

Chapter 3

FEATURE RANKING FOR DEMENTIA DETECTION

While various type of features have been used for dementia detection, it is still unclear which features or their combinations are more effective. We analyze a diverse set of features extracted from spoken language and select the most discriminative ones for dementia detection. We propose two deep-learning-based FR methods called dual dropout ranking (DDR) and dual-net feature ranking (DFR) to rank and select the features.

3.1 Dual Dropout Ranking (DDR)

The proposed DDR is based on a dual-net architecture that separates FS and dementia detection into two neural networks (namely, the operator and selector). The operator is trained on features obtained from the selector to reduce classification/regression loss. The selector is optimized to predict the operator's performance based on automatic regularization. In particular, the selector has dropout masks in its input layer for which the trainable dropout rates are inversely proportional to the features' importance.

3.1.1 Dropout for Feature Ranking

FR aims to rank the importance of individual features according to some criteria, where the criteria typically reflect the features' contributions to the learning performance [3].

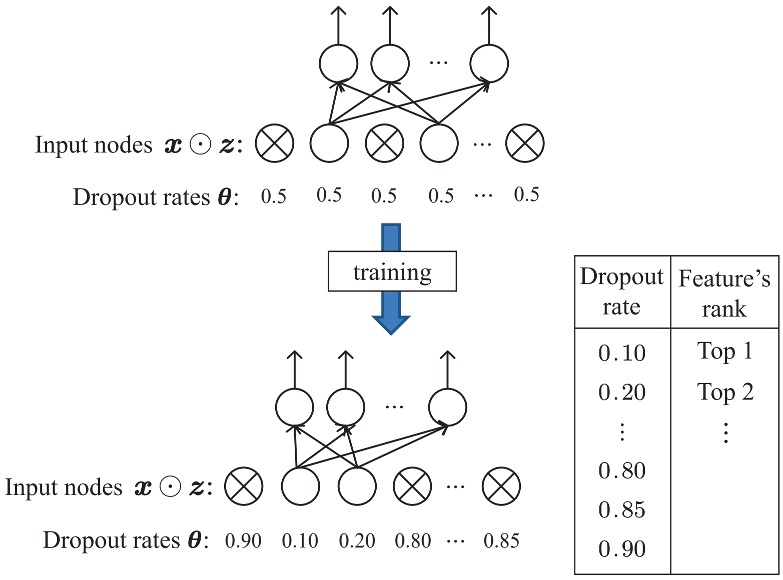
In dropout [71], nodes are purged according to their dropout rates. Therefore, the *higher* the dropout rate, the *lower* the importance of the feature, and FR amounts to determining the dropout rates of individual input nodes. To formulate the dropout rate of a feature, we adopt an approach similar to DropoutFR [4]. Specifically, given a dropout rate vector $\boldsymbol{\theta} = (\theta_1, \theta_2, \ldots, \theta_k, \ldots, \theta_d)$ and a dropout mask vector $\boldsymbol{z} = (z_1, z_2, \ldots, z_k, \ldots, z_d)$, we denote the distribution of $\boldsymbol{z}$ as $q(\boldsymbol{z}) = \prod_{k=1}^{d} q\left(z_k \mid \theta_k\right) = \prod_{k=1}^{d} \operatorname{Bern}\left(z_k \mid \theta_k\right)$, where θ_k is the dropout rate for the k^{th} feature and $z_k \in \{0,1\}$ is the corresponding dropout mask. This gives us a fully factorized Bernoulli distribution that focuses on FR. Suppose $\boldsymbol{x} = (x_1, x_2, \ldots, x_k, \ldots, x_d)$ is an input feature vector. During the forward pass, we place the dropout mask vector on the input layer, that is $\boldsymbol{x} \odot \boldsymbol{z}$, where $\odot$ is the element-wise product (Hadamard product).

3.1.2 Trainable Dropout Rates

In ordinary dropout, the dropout rates are fixed hyper-parameters. Instead of fixing the dropout rates, we treat them as *trainable* parameters. To optimize the dropout rates, we relax the binary dropout masks to *soft* dropout masks as follows:

$$\boldsymbol{z}(\boldsymbol{\theta}) = \operatorname{sigmoid}\left(\frac{1}{t}\left[\log \boldsymbol{\theta} - \log(\mathbf{1} - \boldsymbol{\theta}) + \log \boldsymbol{u} - \log(\mathbf{1} - \boldsymbol{u})\right]\right), \tag{3.1}$$

where $\boldsymbol{u} \in \mathbb{R}^d$ follows the Uniform$(\mathbf{0}, \mathbf{1})$ distribution and t is a normalization constant, which is set to 0.1 in our experiments. Note that this relaxation has also been used in Concrete Dropout [72] and DropoutFR [4]. Eq. (3.1) suggests that $q(\boldsymbol{z})$ places most of the mass to either $z_k = 0$ or $z_k = 1$ to closely resemble the binary dropout mask. With the continuous relaxation in Eq. (3.1), the dropout rates can be optimized through back propagation, and we can gradually select the optimal features $\boldsymbol{x} \odot \boldsymbol{z}$ along with the optimization of the dropout rates. The relation between the features' ranks and trainable dropout rates is depicted in Figure 3.1.


Input nodes $\boldsymbol{x} \odot \boldsymbol{z}$:
Dropout rates $\boldsymbol{\theta}$: 0.5 0.5 0.5 0.5 ⋯ 0.5
training
Input nodes $\boldsymbol{x} \odot \boldsymbol{z}$:
Dropout rates $\boldsymbol{\theta}$: 0.90 0.10 0.20 0.80 ⋯ 0.85

Dropout rate	Feature's rank
0.10	Top 1
0.20	Top 2
⋮	⋮
0.80	
0.85	
0.90	

Figure 3.1: The relationship between the features' ranks and the trainable dropout rates. Before training, each of the input features is assigned the same dropout rate (e.g., 0.5). After training, the features with a lower dropout rate will be assigned a higher rank.

3.1.3 Learning Algorithm

Suppose $\mathcal{M} = \{\mathcal{X}, \mathcal{Y}\}$ is a mini-batch comprising $|\mathcal{M}|$ pairs of $\boldsymbol{x}$ and $\boldsymbol{y}$, where $\boldsymbol{x} \in \mathcal{X}$ is a feature vector of size d, and $\boldsymbol{y} \in \mathcal{Y}$ is the corresponding target. By sampling the uniform distribution in Eq. (3.1), we obtain several *soft* dropout mask vectors $\boldsymbol{z} = (z_1, z_2, \ldots, z_k, \ldots, z_d)$ and form a dropout mask subset $\mathcal{Z}$ of size $|\mathcal{Z}|$. The learning objectives of the dual-net for DDR are defined as:

Operator's objective:[1]

$$\mathcal{L}_O\left(\mathcal{M}, \mathcal{Z}; \psi\right) = \frac{1}{|\mathcal{Z}||\mathcal{M}|}\sum_{\boldsymbol{z}\in\mathcal{Z}} \sum_{(\boldsymbol{x},\boldsymbol{y})\in\mathcal{M}} l(\boldsymbol{x}\odot\boldsymbol{z}, \boldsymbol{y}; \psi) \tag{3.2a}$$

Selector's objective:[2]

$$\mathcal{L}_S\left(\mathcal{Z}(\boldsymbol{\theta}); \varphi\right) = \frac{1}{|\mathcal{Z}|}\sum_{\boldsymbol{z}\in\mathcal{Z}} \left\{ \left| f_S(\boldsymbol{z}; \varphi) - \frac{1}{|\mathcal{M}|}\sum_{(\boldsymbol{x},\boldsymbol{y})\in\mathcal{M}} l(\boldsymbol{x}\odot\boldsymbol{z}, \boldsymbol{y}; \psi)\right| \Bigg/ \sum_{k=1}^{d}(1 - z_k) \right\} \tag{3.2b}$$

where $l(\boldsymbol{x}\odot\boldsymbol{z}, \boldsymbol{y}; \psi)$ is either the cross-entropy loss for binary/multi-class classification or the MSE loss for regression, ψ is the operator's parameters, $f_S(\boldsymbol{z}, \varphi)$ is the selector's output, and φ is the selector's parameters. The relationship between the operator and the selector in the dual-net architecture is depicted in Figure 3.2. During training, the operator and selector are trained alternately. The alternate training procedure is depicted in Appendix 1. The advantages of the dual-net architecture are as follows. 1) It can off-load the optimization of dropout rates to the selector, which lets the operator to focus on the classification or regression tasks. 2) It can shift the FS constraint (the denominator of Eq. (3.2b)) to the selector, and with the alternate training procedure, it enables *automatic regularization*. 3) It avoids manually setting the regularization coefficients.

Operator

The operator is trained on the features selected by the selector to reduce classification loss. For each iteration, given the dropout mask subset $\mathcal{Z}$ from the selector, the

[1] During the optimization of the operator, $\boldsymbol{\theta}$ is considered fixed. Therefore, we drop the dependence of $\boldsymbol{z}$ on $\boldsymbol{\theta}$.

[2] For notational simplicity, we omit the dependence of $\boldsymbol{z}$ on $\boldsymbol{\theta}$ on the right side of this equation.

selected features $\{\boldsymbol{x} \odot \boldsymbol{z}\}_{\boldsymbol{x} \in \mathcal{X}, \boldsymbol{z} \in \mathcal{Z}}$ are fed to the operator, and the operator's learning performance based on the selected features is obtained. Given the selected features $\boldsymbol{x} \odot \boldsymbol{z}$, $\frac{1}{|\mathcal{M}|} \sum_{(\boldsymbol{x},\boldsymbol{y}) \in \mathcal{M}} l(\boldsymbol{x} \odot \boldsymbol{z}, \boldsymbol{y}; \psi)$ is the learning performance of the operator on the mini-batch $\mathcal{M}$. By enumerating $\boldsymbol{z}$ in $\mathcal{Z}$, we obtain the average learning performance of the operator on the mini-batch. Then, we update the operator's parameters and pass the operator's learning performance to the selector as a feedback indicating how well the operator performs on the selected features. Different from the sparsity regularization methods that also incorporate the regularization into the network, the operator only focuses on reducing classification loss. Given the selected features, the operator's architecture can be tailored to different learning tasks (classification or regression).

Selector

The selector learns to predict the operator's learning performance using as few selected features as possible. The mean absolute error (MAE) between $f_S(\boldsymbol{z}, \varphi)$ and $\frac{1}{|\mathcal{M}|} \sum_{(\boldsymbol{x},\boldsymbol{y}) \in \mathcal{M}} l(\boldsymbol{x} \odot \boldsymbol{z}, \boldsymbol{y}; \psi)$ requires that the selector closely predicts the operator's learning performance. The constraint $\sum_{k=1}^{d} (1 - z_k)$ on the denominator of Eq. (3.2b) automatically causes most of the dropout masks in $\boldsymbol{z}$ to become 0; so the selector only selects a small number of features when predicting the operator's learning performance.

After training and updating the selector's parameters and dropout rates, we have the updated dropout rate vector $\boldsymbol{\theta}'$. Through sampling the uniform distribution in Eq. (3.1), we obtain several new soft dropout mask vectors $\boldsymbol{z}'$ from the updated dropout rate vector $\boldsymbol{\theta}'$ and form a new dropout mask subset $\mathcal{Z}'$ for the next iteration. In practical implementation, the dropout mask vector fed to the selector is $\boldsymbol{z} \odot \boldsymbol{z}'$, where $\boldsymbol{z} \in \mathcal{Z}$ and $\boldsymbol{z}' \in \mathcal{Z}'$.

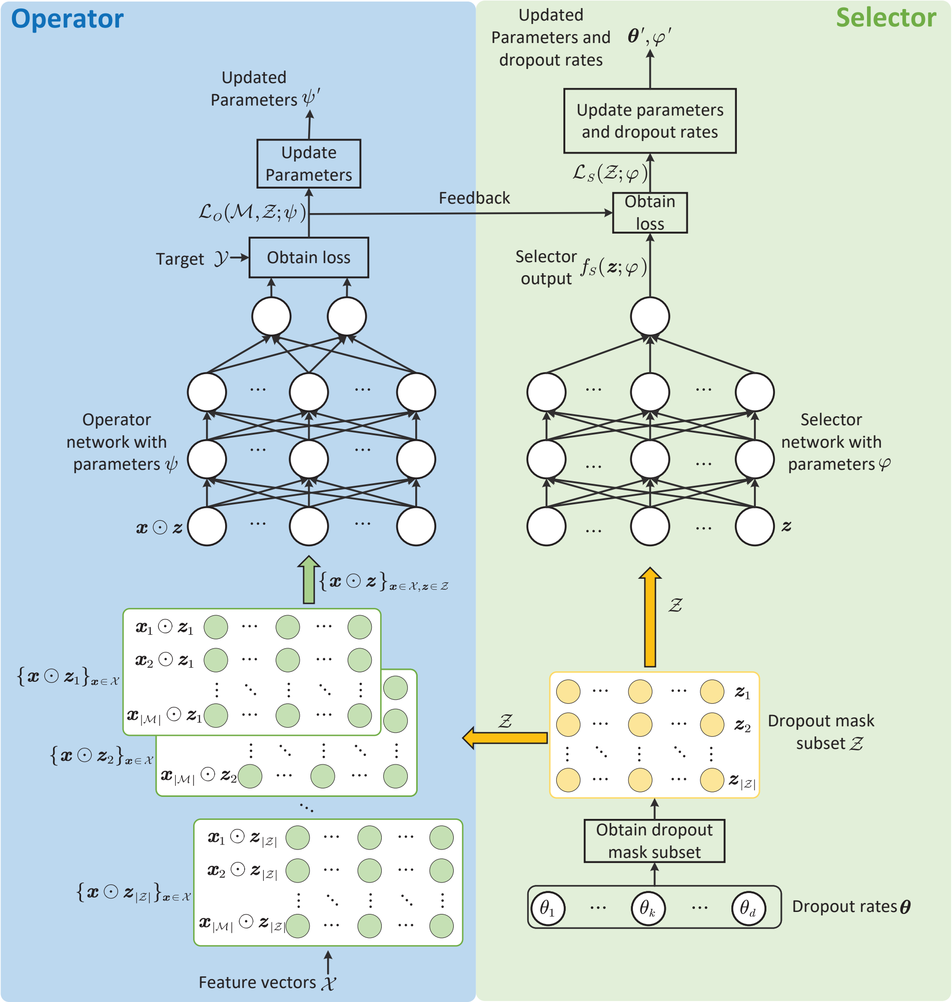

Operator
Selector
Updated Parameters ψ'
Update Parameters
$\mathcal{L}_O(\mathcal{M},\mathcal{Z};\psi)$
Feedback
Target $\mathcal{Y}$
Obtain loss
Operator network with parameters ψ
$\boldsymbol{x}\odot\boldsymbol{z}$
$\{\boldsymbol{x}\odot\boldsymbol{z}\}_{\boldsymbol{x}\in\mathcal{X},\boldsymbol{z}\in\mathcal{Z}}$
$\{\boldsymbol{x}\odot\boldsymbol{z}_1\}_{\boldsymbol{x}\in\mathcal{X}}$
$\boldsymbol{x}_1\odot\boldsymbol{z}_1$
$\boldsymbol{x}_2\odot\boldsymbol{z}_1$
$\boldsymbol{x}_{|\mathcal{M}|}\odot\boldsymbol{z}_1$
$\{\boldsymbol{x}\odot\boldsymbol{z}_2\}_{\boldsymbol{x}\in\mathcal{X}}$
$\boldsymbol{x}_{|\mathcal{M}|}\odot\boldsymbol{z}_2$
$\{\boldsymbol{x}\odot\boldsymbol{z}_{|\mathcal{Z}|}\}_{\boldsymbol{x}\in\mathcal{X}}$
$\boldsymbol{x}_1\odot\boldsymbol{z}_{|\mathcal{Z}|}$
$\boldsymbol{x}_2\odot\boldsymbol{z}_{|\mathcal{Z}|}$
$\boldsymbol{x}_{|\mathcal{M}|}\odot\boldsymbol{z}_{|\mathcal{Z}|}$
Feature vectors $\mathcal{X}$
Updated Parameters and dropout rates $\boldsymbol{\theta}',\varphi'$
Update parameters and dropout rates
$\mathcal{L}_S(\mathcal{Z};\varphi)$
Obtain loss
Selector output $f_S(\boldsymbol{z};\varphi)$
Selector network with parameters φ
$\boldsymbol{z}$
$\mathcal{Z}$
$\mathcal{Z}$
$\boldsymbol{z}_1$
$\boldsymbol{z}_2$
$\boldsymbol{z}_{|\mathcal{Z}|}$
Dropout mask subset $\mathcal{Z}$
Obtain dropout mask subset
θ_1 ⋯ θ_k ⋯ θ_d
Dropout rates $\boldsymbol{\theta}$

Figure 3.2: The dual-net architecture of DDR. ψ and φ represent the network parameters of the operator and selector, respectively. $\boldsymbol{\theta}$ comprises the dropout rates at the input layer of the selector. $\mathcal{X}$ contains $|\mathcal{M}|$ feature vectors and $\mathcal{Z}$ contains $|\mathcal{Z}|$ dropout masks.

3.1.4 Two-step Feature Selection

In this section, we extend the DDR in Section 3.1.3 to a *two-step FS* approach, which aims to deal with the circumstance where the feature dimensions are much larger than the number of training samples. We present a two-step FS method – Step 1 utilizes filter methods to pre-screen features; and Step 2 uses DDR to rank the screened features and for selecting spoken language biomarkers. The two-step FS method is depicted in Figure 3.3. FS can be nested inside CV, which means that FS is conducted on the TR of individual folds instead of the entire training set. Filter methods are usually computationally cheap and do not require training. When the feature dimension is very high, filter methods are indispensable for obtaining a reduced set of features for the expensive FS methods. Therefore, in Step 1, filter methods are utilized to pre-screen the original features. Three filter methods were evaluated in the experiments: FDR [63], PeaCorr tests, and MutInfo. In Step 2, the proposed DDR is applied to rank the remaining features. Before training, each of the remaining features is assigned the same dropout rate (e.g., 0.5). During training, the DDR adjusts the dropout rates to reflect the features' importance. After training, we rank the features according to the dropout rates and select the features with low dropout rates.

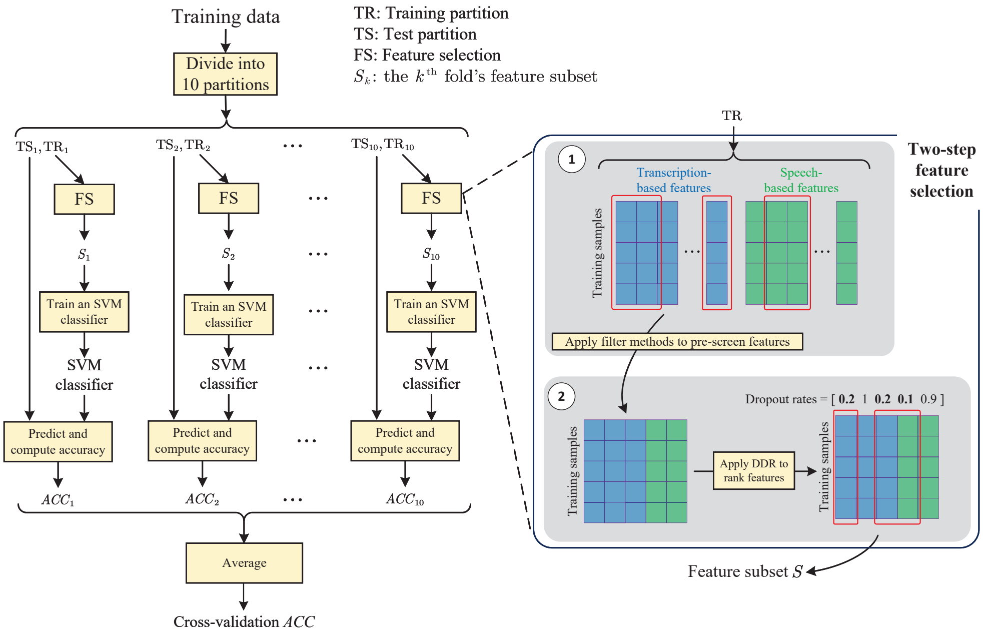

Training data
Divide into 10 partitions
TR: Training partition
TS: Test partition
FS: Feature selection
S_k: the k^{th} fold's feature subset
TS_1, TR_1
TS_2, TR_2
$\text{TS}_{10}, \text{TR}_{10}$
FS
S_1
S_2
S_{10}
Train an SVM classifier
SVM classifier
Predict and compute accuracy
ACC_1
ACC_2
ACC_{10}
Average
Cross-validation ACC
TR
Two-step feature selection
1
Transcription-based features
Speech-based features
Training samples
Apply filter methods to pre-screen features
2
Dropout rates = [0.2 1 0.2 0.1 0.9]
Apply DDR to rank features
Feature subset S

Figure 3.3: The FS procedure: 10-fold CV was adopted. The training data were divided into 10 TR. In the TR of individual folds, the two-step FS approach was applied to select the most discriminative features. In Step 1, filter methods were utilized to pre-screen the features. In Step 2, DDR was adopted to rank the features selected in Step 1 (the remaining features). Features with low dropout rates were then selected.

3.1.5 Feature Selection in Cross-validation

As illustrated in Figure 3.3, an approach termed *nested* FS is adopted within the 10-fold CV as opposed to the traditional approach to conducting FS outside the CV. Specifically, within each fold of the CV, the TR is used to select features, which are then tested using the TS. As the training data differ among the individual folds, different features will be selected in each fold. Conducting FS on the entire training data prior to CV introduces bias, as the TS will also be used for FS. This may ultimately affect the CV results.

3.2 Dual-net Feature Ranking

In this section, we first explain why the parameters of a linear regression model can determine feature relevancy. Then, we extend the concept of feature relevancy determination to deep neural networks and proposed a deep-learning-based FR method called dual-net feature ranking (DFR). The method utilizes a dual-net architecture, where two networks (called operator and selector) are trained to simultaneously perform FS and dementia detection. Specifically, the selector is trained to find multiple subsets of features to predict the operator's performance, and the operator uses these feature subsets to minimize classification errors. DFR uses all of the selector's parameters to determine the contributions of individual features to the selector's predictions. Specifically, we summarize our main contributions as follows.

1. We introduce a novel approach for interpreting the contribution of the input variables to the neural network's output. The approach utilizes the parameters of the network to interpret the contribution of the input variables, taking into account the *non-linear* relationships between the input variables and the network's output. It enables assessing the contribution of individual input variables in a *multi-layer* neural network.

2. We propose a feature ranking method based on a dual-net architecture, consisting of an operator net and a selector net. The selector net always has *one* linear output node, enabling it to interpret the contribution of individual input variables. On the other hand, the operator net could have *multiple* output nodes, making it suitable for classification.

3.2.1 Variable Selection in Deep Neural Networks

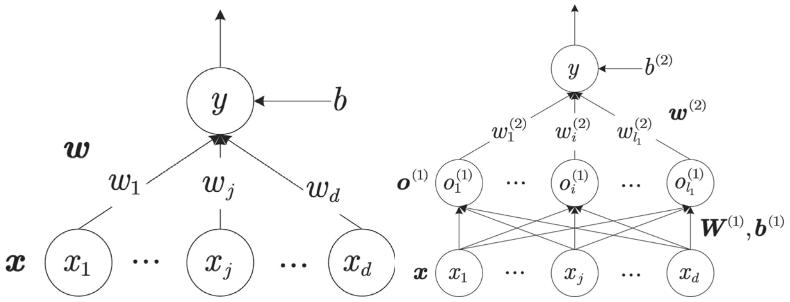
We consider the usual *linear regression* model. Given d predictor variables $\boldsymbol{x} = (x_1, \dots, x_j, \dots, x_d)^\mathsf{T}$, the response variable $f(\boldsymbol{x})$ is predicted by

$$f(\boldsymbol{x}) \approx \hat{f}(\boldsymbol{x}) = \hat{\beta}_0 + \hat{\beta}_1 x_1 + \cdots + \hat{\beta}_j x_j + \cdots + \hat{\beta}_d x_d, \tag{3.3}$$

where $\hat{f}(\boldsymbol{x})$ is a linear model and $\hat{\beta}_1, \dots, \hat{\beta}_j, \dots, \hat{\beta}_d$ are its parameters. Since there is a one-to-one correspondence between the model parameters and the predictor variables, the effect of a given predictor x_j on the model $\hat{f}(\boldsymbol{x})$ can be evaluated through the value of $\hat{\beta}_j$ [70].[3] In particular, when the model parameter $\hat{\beta}_j \gg 0$, the predictor x_j may have a significant positive effect on the model. When $\hat{\beta}_j \approx 0$, we may say that x_j contributes little to the prediction of $f(\boldsymbol{x})$, and it can be removed from the model.

Alhanai *et al.* [56] determined the most predictive features for mild cognitive impairment (MCI) detection by evaluating the values of the corresponding parameters in a logistic-regression model. They found that decreasing pitch, decreasing jitter, and shorter speech segment lengths are positively correlated with MCI. Following this strategy, we first introduce the variable selection in a 1-layer fully-connected network with one linear output node because it is equivalent to the linear regression model. We then extend the strategy to a multi-layer fully-connected network.

[3] The model's parameters are standardized so that the relevance of the predictor variables can be meaningfully compared.



(a) (b)

Figure 3.4: Network architectures for explaining the contributions of individual features (input nodes) to the network's prediction. (a) A 1-layer network with one linear output is equivalent to the linear regression model. (b) Feature importance can be obtained from the weights of a multi-layer network. See Section 3.2.1 for details.

A 1-layer fully-connected network with one linear output node (Figure 3.4(a)) is equivalent to the linear regression model in Eq. (3.3). Given a d-dimensional input vector $\boldsymbol{x} = (x_1, \ldots, x_j, \ldots, x_d)^\intercal$, the network's output y is (omitting the bias for simplicity):

$$y = \boldsymbol{w}^\intercal \boldsymbol{x} = w_1 x_1 + \cdots + w_j x_j + \cdots + w_d x_d, \tag{3.4}$$

where $\boldsymbol{w} = (w_1, \ldots, w_j, \ldots, w_d)^\intercal$ is the network's weight vector. As Eq. (3.4) is equivalent to Eq. (3.3), we can also explain the effect of a given input variable x_j on the prediction of the network through the value of w_j. In particular, when $w_j \gg 0$, we may say that x_j has a significant positive effect on the network's output.[4] When $w_j \approx 0$, we may say that x_j is irrelevant to the network's output and can be removed from the network. We formulate a one-to-one correspondence between the input $\boldsymbol{x}$ and the network's weight vector $\boldsymbol{w}$:

$$\text{diag}\{\boldsymbol{x}\}\boldsymbol{w} = (w_1 x_1, \ldots, w_j x_j, \ldots, w_d x_d)^\intercal, \tag{3.5}$$

[4] We explain why considering positive weights in Section 3.2.3.

where diag$\{\boldsymbol{x}\}$ is a diagonal matrix with $\{x_j\}$ in its diagonal. By setting $\boldsymbol{x} = \mathbf{1}$ in Eq. (3.5), we obtain a *feature importance vector* $\boldsymbol{c}$:

$$\boldsymbol{c} = \text{diag}\{\mathbf{1}\}\boldsymbol{w} = \boldsymbol{w} = (w_1, \ldots, w_j, \ldots, w_d)^{\mathsf{T}}. \tag{3.6}$$

Eq. (3.6) suggests that the bigger the value of w_j, the more important the input variable x_j. Therefore, we can select the important input variables according to $\boldsymbol{c}$.

Figure 3.4(b) depicts a 2-layer network with the hidden layer having l_1 nodes and the output layer having one node. Suppose $\boldsymbol{W}^{(1)}$ is a $d \times l_1$ weight matrix connecting the input $\boldsymbol{x}$ to the hidden layer and $\boldsymbol{b}^{(1)} \in \mathbb{R}^{l_1}$ is the corresponding bias vector. Also, suppose $\boldsymbol{w}^{(2)} = (w_1^{(2)}, \ldots, w_i^{(2)}, \ldots, w_{l_1}^{(2)})^{\mathsf{T}}$ is the weight vector of the output layer and $b^{(2)}$ is the bias. Given a d-dimensional input vector $\boldsymbol{x}$, the output of the hidden layer is (omitting the bias for simplicity):

$$\boldsymbol{o}^{(1)} = g\left(\left(\boldsymbol{W}^{(1)}\right)^{\mathsf{T}} \boldsymbol{x}\right) \in \mathbb{R}^{l_1}, \tag{3.7}$$

where $g(\cdot)$ is a non-linear activation function, e.g., sigmoid. And the output of the network is:

$$y = (\boldsymbol{w}^{(2)})^{\mathsf{T}} \boldsymbol{o}^{(1)} = (\boldsymbol{w}^{(2)})^{\mathsf{T}} g\left(\left(\boldsymbol{W}^{(1)}\right)^{\mathsf{T}} \boldsymbol{x}\right). \tag{3.8}$$

Comparing Eq. (3.4) and Eq. (3.8) and following Eq. (3.5), we can also formulate a one-to-one correspondence between the input $\boldsymbol{x}$ and the network's parameters:

$$g\left(\text{diag}\{\boldsymbol{x}\}\boldsymbol{W}^{(1)}\right) \boldsymbol{w}^{(2)} = \left[v_1(x_1), \ldots, v_j(x_j), \ldots, v_d(x_d)\right]^{\mathsf{T}}, \tag{3.9}$$

where $v_j(x_j) = \sum_{k=1}^{l_1} g\left(w_{j,k}^{(1)} x_j\right) w_k^{(2)}$.[5] Again, by setting $\boldsymbol{x} = \mathbf{1}$, we can obtain the

[5] The clarification of the one-to-one correspondence is shown in Section 3.2.3.

feature importance vector:

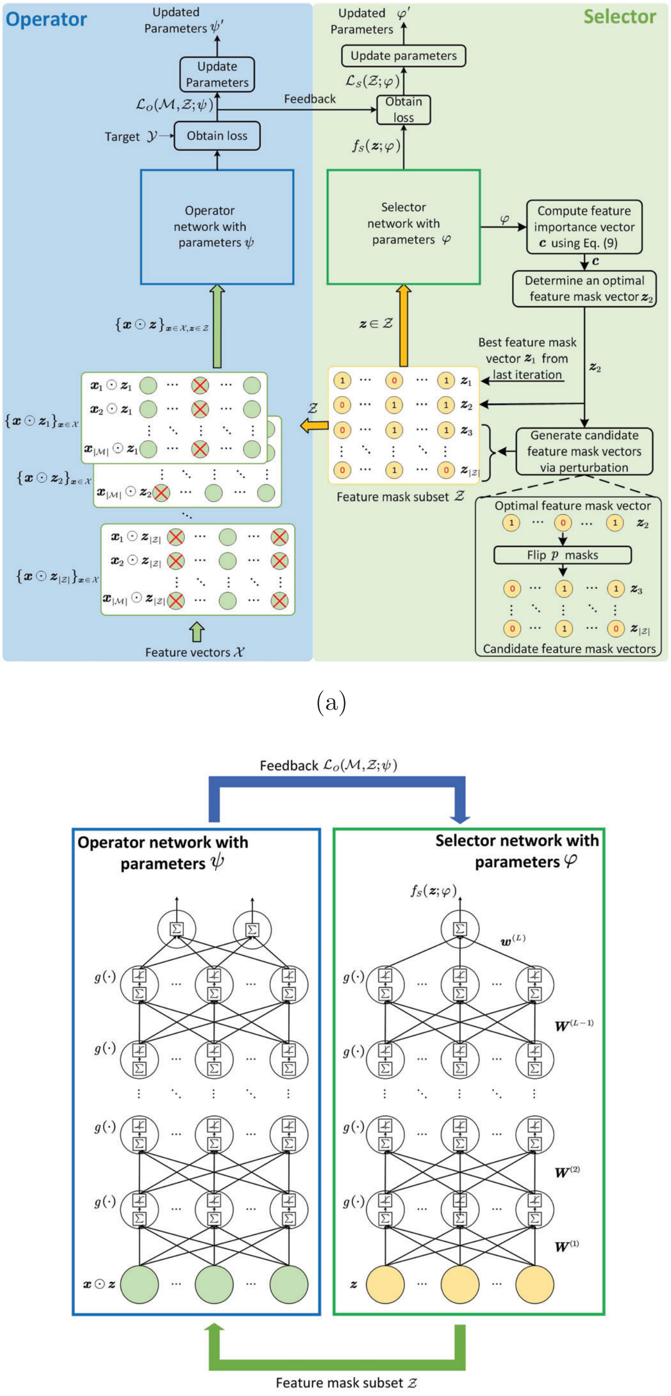
$$\boldsymbol{c} = g\left(\mathrm{diag}\{\mathbf{1}\}\boldsymbol{W}^{(1)}\right)\boldsymbol{w}^{(2)} = g\left(\boldsymbol{W}^{(1)}\right)\boldsymbol{w}^{(2)} \in \mathbb{R}^d. \tag{3.10}$$

Note that $\boldsymbol{c}$ is also a d-dimensional vector with c_j corresponding to the input variable x_j. Similar results can be extended to an L-layer neural network with weight matrices $\{\boldsymbol{W}^{(i)}, i = 1, 2, \ldots, L-1\}$ for the hidden layers and weight vector $\boldsymbol{w}^{(L)}$ for the output layer. The feature importance vector $\boldsymbol{c}$ for the L-layer network is:

$$\boldsymbol{c} = g\left(g\left(g\left(\boldsymbol{W}^{(1)}\right)\boldsymbol{W}^{(2)}\right)\cdots\boldsymbol{W}^{(L-1)}\right)\boldsymbol{w}^{(L)} \in \mathbb{R}^d. \tag{3.11}$$

3.2.2 Learning Algorithm

In Section 3.2.1, we formulate a d-dimensional feature importance vector $\boldsymbol{c}$ that reflects the feature importance of the input variables. We use $\boldsymbol{c}$ to determine the contribution of the input variables to the output of a deep neural network. Specifically, the input variable x_j with a larger c_j will have a greater contribution to the output. Based on the feature importance vector $\boldsymbol{c}$, we propose a deep-learning-based FS method called DFR. DFR comprises two deep neural networks (called operator and selector), as shown in Figure 3.5. During training, the operator and selector are trained alternately. The alternate learning algorithm is depicted in Algorithm 2.


Operator
Selector
Updated Parameters ψ'
Update Parameters
$\mathcal{L}_O(\mathcal{M},\mathcal{Z};\psi)$
Feedback
Target $\mathcal{Y}$
Obtain loss
Operator network with parameters ψ
$\{\boldsymbol{x}\odot\boldsymbol{z}\}_{\boldsymbol{x}\in\mathcal{X},\boldsymbol{z}\in\mathcal{Z}}$
$\{\boldsymbol{x}\odot\boldsymbol{z}_1\}_{\boldsymbol{x}\in\mathcal{X}}$
$\{\boldsymbol{x}\odot\boldsymbol{z}_2\}_{\boldsymbol{x}\in\mathcal{X}}$
$\{\boldsymbol{x}\odot\boldsymbol{z}_{|\mathcal{Z}|}\}_{\boldsymbol{x}\in\mathcal{X}}$
Feature vectors $\mathcal{X}$
Updated φ' Parameters
Update parameters
$\mathcal{L}_S(\mathcal{Z};\varphi)$
Obtain loss
$f_S(\boldsymbol{z};\varphi)$
Selector network with parameters φ
φ
Compute feature importance vector $\boldsymbol{c}$ using Eq. (9)
$\boldsymbol{c}$
Determine an optimal feature mask vector $\boldsymbol{z}_2$
$\boldsymbol{z}\in\mathcal{Z}$
Best feature mask vector $\boldsymbol{z}_1$ from last iteration
$\boldsymbol{z}_2$
$\mathcal{Z}$
Feature mask subset $\mathcal{Z}$
Generate candidate feature mask vectors via perturbation
Optimal feature mask vector
Flip p masks
Candidate feature mask vectors
Feedback $\mathcal{L}_O(\mathcal{M},\mathcal{Z};\psi)$
Operator network with parameters ψ
Selector network with parameters φ
$f_S(\boldsymbol{z};\varphi)$
$g(\cdot)$
$\boldsymbol{w}^{(L)}$
$\boldsymbol{W}^{(L-1)}$
$\boldsymbol{W}^{(2)}$
$\boldsymbol{W}^{(1)}$
$\boldsymbol{x}\odot\boldsymbol{z}$
$\boldsymbol{z}$
Feature mask subset $\mathcal{Z}$

(a)

(b)

Figure 3.5: The training procedure (a) and block diagram (b) of DFR. The selector network is a multi-layer fully-connected network with one linear output node. At the beginning of each iteration, the selector's parameters $\varphi = \{\boldsymbol{W}^{(1)}, \boldsymbol{W}^{(2)}, \ldots, \boldsymbol{W}^{(L-1)}, \boldsymbol{w}^{(L)}\}$ are used to compute the feature importance vector $\boldsymbol{c}$. $g(\cdot)$ is the sigmoid activation function.

Suppose $\mathcal{M} = \{\mathcal{X}, \mathcal{Y}\}$ is a mini-batch comprising $|\mathcal{M}|$ pairs of $\boldsymbol{x}$ and $\boldsymbol{y}$, where $\boldsymbol{x} \in \mathcal{X}$ is a feature vector of size d, and $\boldsymbol{y} \in \mathcal{Y}$ is the corresponding target. The learning algorithm of DFR is defined in Eq. (3.12), where $\mathcal{L}_O\left(\mathcal{M}, \mathcal{Z}; \psi\right)$ is the operator's objective, $l(\boldsymbol{x} \odot \boldsymbol{z}, \boldsymbol{y}; \psi)$ is either the cross-entropy loss for classification or the MSE loss for regression, and ψ denotes the operator's parameters. $\mathcal{L}_S\left(\mathcal{Z}; \varphi\right)$ is the selector's objective, $f_S(\boldsymbol{z}, \varphi)$ is the selector's output, and $\varphi = \{\boldsymbol{W}^{(1)}, \boldsymbol{W}^{(2)}, \ldots, \boldsymbol{W}^{(L-1)}, \boldsymbol{w}^{(L)}\}$ contains the selector's parameters.

Operator's objective:

$$\mathcal{L}_O\left(\mathcal{M}, \mathcal{Z}; \psi\right) = \frac{1}{|\mathcal{Z}||\mathcal{M}|} \sum_{\boldsymbol{z} \in \mathcal{Z}} \sum_{(\boldsymbol{x}, \boldsymbol{y}) \in \mathcal{M}} l(\boldsymbol{x} \odot \boldsymbol{z}, \boldsymbol{y}; \psi) \tag{3.12a}$$

Selector's objective:

$$\mathcal{L}_S\left(\mathcal{Z}; \varphi\right) = \frac{1}{|\mathcal{Z}|} \sum_{\boldsymbol{z} \in \mathcal{Z}} \left\{ \left| f_S(\boldsymbol{z}; \varphi) - \frac{1}{|\mathcal{M}|} \sum_{(\boldsymbol{x}, \boldsymbol{y}) \in \mathcal{M}} l(\boldsymbol{x} \odot \boldsymbol{z}, \boldsymbol{y}; \psi) \right| \right\} \tag{3.12b}$$

Operator

The operator is trained on the features selected by the selector to reduce the loss $\mathcal{L}_O\left(\mathcal{M}, \mathcal{Z}; \psi\right)$. The feature mask vector $\boldsymbol{z}$ in the feature mask subset $\mathcal{Z}$ indicates which features have been selected. For each iteration, given the feature mask subset $\mathcal{Z}$ from the selector, the selected features $\{\boldsymbol{x} \odot \boldsymbol{z}\}_{\boldsymbol{x} \in \mathcal{X}, \boldsymbol{z} \in \mathcal{Z}}$ are fed to the operator, and the operator's learning performance based on the selected features is obtained. Given the selected features $\boldsymbol{x} \odot \boldsymbol{z}$, $\frac{1}{|\mathcal{M}|} \sum_{(\boldsymbol{x}, \boldsymbol{y}) \in \mathcal{M}} l(\boldsymbol{x} \odot \boldsymbol{z}, \boldsymbol{y}; \psi)$ is the learning performance of the operator on the mini-batch $\mathcal{M}$. Then, we pass the operator's learning performance to the selector as a feedback indicating how well the operator performs on the selected features.

Selector

The selector learns to predict the operator's learning performance using the selected features. The MAE between $f_S(\boldsymbol{z}, \varphi)$ and $\frac{1}{|\mathcal{M}|}\sum_{(\boldsymbol{x},\boldsymbol{y})\in\mathcal{M}} l(\boldsymbol{x} \odot \boldsymbol{z}, \boldsymbol{y}; \psi)$ requires that the selector accurately predicts the operator's learning performance. At the beginning of each iteration, the selector produces the feature mask subset $\mathcal{Z}$ using the following steps:

(i) *Retain the best feature mask vector.* We retain the best feature mask vector $\boldsymbol{z}_1$ that achieves the best learning performance (e.g., the smallest cross-entropy loss) in the last iteration.[6]

(ii) *Determine an optimal feature mask vector.* We compute the feature importance vector $\boldsymbol{c}$ using Eq. (3.11) based on the selector's parameters φ. According to the feature importance vector $\boldsymbol{c}$, we generate an optimal feature mask vector $\boldsymbol{z}_2$ by assigning the top s features with mask 1 and the rest of $d - s$ features with mask 0.

(iii) *Generate candidate feature mask vectors.* To increase the diversity of the feature mask vectors, we generate several candidate feature mask vectors $\{\boldsymbol{z}_3, \ldots, \boldsymbol{z}_{|\mathcal{Z}|}\}$ by randomly flipping p masks in $\boldsymbol{z}_2$.

(iv) *Produce the feature mask subset.* Finally, we produce the feature mask subset $\mathcal{Z} = \{\boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3, \ldots, \boldsymbol{z}_{|\mathcal{Z}|}\}$.

3.2.3 Efficiency of the Learning Algorithm

We first demonstrate that the parameters of the selector net can be used to evaluate the contributions of input variables to the the network output. We illustrate this

[6] In the first iteration, $\boldsymbol{z}_1$ is randomly initialized.

using an example in which the selector net consists of only one layer, as shown in Figure 3.6.

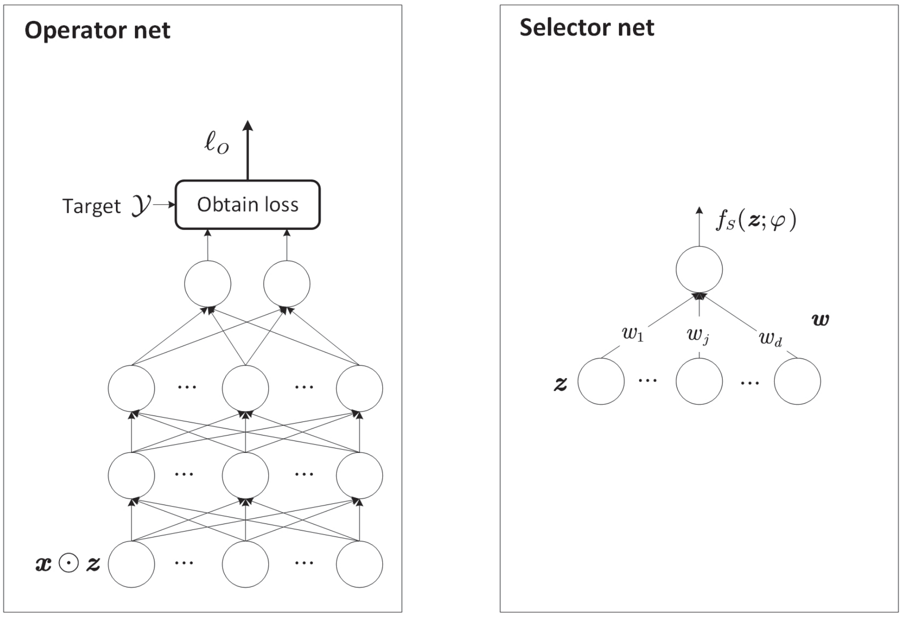


Figure 3.6: An example of our dual-net architecture, in which the selector net consists of only one layer. $\boldsymbol{z} = (z_1, ..., z_j, ..., z_d)^{\mathsf{T}}$ is the feature marks. $\varphi = \{\boldsymbol{w}\}$ is the selector net's parameters.

Given the feature marks $\boldsymbol{z} = (z_1, ..., z_j, ..., z_d)^{\mathsf{T}}$, the output of the selector net is (omitting the bias for simplicity) is:

$$f_S(\boldsymbol{z};\varphi) = w_1 z_1 + \cdots + w_j z_j + \cdots + w_d z_d. \tag{3.13}$$

Given the feature marks $\boldsymbol{z}$ and input variables $\boldsymbol{x}$, the operator's learning performance is:

$$\ell_O = \frac{1}{|\mathcal{M}|} \sum_{(\boldsymbol{x},\boldsymbol{y})\in\mathcal{M}} l(\boldsymbol{x} \odot \boldsymbol{z}, \boldsymbol{y}; \psi), \tag{3.14}$$

where $\mathcal{M} = \{\mathcal{X}, \mathcal{Y}\}$ is a mini-batch comprising $|\mathcal{M}|$ pairs of $\boldsymbol{x}$ and $\boldsymbol{y}$, and ψ represents the operator net's parameters. As described before, the selector learns to predict the operator's learning performance using the selected features, and the selector loss is:

$$\ell_S = |f_S(\boldsymbol{z}; \varphi) - \ell_O| = |w_1 z_1 + \cdots + w_j z_j + \cdots + w_d z_d - \ell_O| \,. \tag{3.15}$$

Given that the gradient for the absolute error loss [73] is:

$$\text{sign}\left[y_{pred} - y_{true}\right] = \begin{cases} +1, \text{if} & y_{pred} > y_{true} \\ -1, \text{if} & y_{pred} < y_{true} \end{cases} . \tag{3.16}$$

When $y_{pred} > y_{true}$, the partial derivatives of ℓ_S with respect to z_j is:[7]

$$\frac{\partial \ell_S}{\partial z_j} = \text{sign}\left[y_{pred} - y_{true}\right] \cdot \frac{\partial f_S(\boldsymbol{z}; \varphi)}{\partial z_j} = +1 \cdot \frac{\partial f_S(\boldsymbol{z}; \varphi)}{\partial z_j} = w_j. \tag{3.17}$$

Computing the partial derivatives with respect to each of the elements in $\boldsymbol{z}$, we have:

$$\left(\frac{\partial \ell_S}{\partial z_1}, \cdots, \frac{\partial \ell_S}{\partial z_j}, \cdots \frac{\partial \ell_S}{\partial z_d}\right)^{\mathsf{T}} = \left(w_1, ..., w_j, ..., w_d\right)^{\mathsf{T}}, \tag{3.18}$$

which is exactly the feature importance vector $\boldsymbol{c}$ in Eq. (3.6). Eq. (3.18) indicates that a unit increase in z_j would increase y_{pred} by an amount w_j. In such cases, we can utilize $\boldsymbol{c} = \left(w_1, ..., w_j, ..., w_d\right)^{\mathsf{T}}$ to measure the contributions of $\boldsymbol{z} = \left(z_1, ..., z_j, ..., z_d\right)^{\mathsf{T}}$ to y_{pred}. In other words, we can utilize the parameters of the selector net to evaluate the contributions of input variables to the network output.

We only consider the features with positive weights in $\boldsymbol{c}$. We provide the reason using a single-layer selector net consisting of one layer, as shown in Figure 3.6. Given the feature masks $\boldsymbol{z} = \left(z_1, ..., z_j, ..., z_d\right)^{\mathrm{T}}$ ($z_j = 0$ or 1) and the selector's weights $\boldsymbol{w} = \left(w_1, ..., w_j, ..., w_d\right)^{\mathrm{T}}$, the output of the selector is (omitting the bias for simplicity):

$$f_S(\boldsymbol{z}; \varphi) = w_1 z_1 + \cdots + w_j z_j + \cdots + w_d z_d. \tag{3.19}$$

[7] When updating the parameters of the selector net, we keep the operator network fixed. Therefore, we treat ℓ_O as a constant independent on $\boldsymbol{z}$.

The selector learns to predict the operator's classification loss (cross entropy). Since cross entropy is always non-negative, we require the selector's prediction to be also non-negative. In this case, if $z_j = 1$, a larger positive weight w_j will have a greater impact on the selector's prediction. If $z_j = 0$, $w_j z_j$ will be 0, and w_j will have no effect on the selector's prediction. We do not select large negative weights because it may render the prediction negative. For example, if the selector's weights are $w_1 = -1.5$, $w_2 = 1.0$, and $w_3 = 0.3$, and we select two features with large positive weights, the selector's prediction will be:

$$\begin{aligned} f_S(\boldsymbol{z}; \varphi) &= w_1 z_1 + w_2 z_2 + w_3 z_3 \\ &= -1.5 \times 0 + 1.0 \times 1 + 0.3 \times 1 \\ &= 1.3 > 0. \end{aligned} \tag{3.20}$$

Conversely, if we select two features with large positive or negative weights, the selector's prediction will be:

$$\begin{aligned} f_S(\boldsymbol{z}; \varphi) &= w_1 z_1 + w_2 z_2 + w_3 z_3 \\ &= -1.5 \times 1 + 1.0 \times 1 + 0.3 \times 0 \\ &= -0.5 < 0, \end{aligned} \tag{3.21}$$

which is unexpected because classification loss is always non-negative. A similar argument applies to multi-layer selector nets.

We then demonstrate that setting some elements in the feature mask vector $\boldsymbol{z}$ to 0 is a regularization approach that leads to coefficient sparsity. We commence with a linear regression model in which some of the inputs are masked, as shown in Figure 3.7.

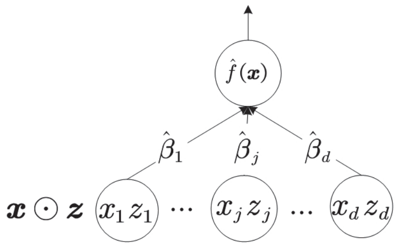


Figure 3.7: A linear regression model in which the inputs $\boldsymbol{x}$ are masked by the feature mask vector $\boldsymbol{z}$.

In Figure 3.7, $\hat{\boldsymbol{\beta}} = \left(\hat{\beta}_1, ..., \hat{\beta}_j, ..., \hat{\beta}_d\right)^{\mathsf{T}}$ denotes the model coefficients, $\boldsymbol{x} = (x_1, ..., x_j, ..., x_d)^{\mathsf{T}}$ represents the input variables, and $\boldsymbol{z} = (z_1, ..., z_j, ..., z_d)^{\mathsf{T}}$ stands for the feature mask vector. The linear regression model takes the following masked inputs:

$$\boldsymbol{x} \odot \boldsymbol{z} = (x_1 z_1, ..., x_j z_j, ..., x_d z_d)^{\mathsf{T}} . \tag{3.22}$$

The regression model produces an output (omitting the bias for simplicity):

$$\begin{aligned} \hat{f}(\boldsymbol{x}) &= \hat{\beta}_1 x_1 z_1 + \cdots + \hat{\beta}_j x_j z_j + \cdots + \hat{\beta}_d x_d z_d \\ &= \hat{\beta}_1 z_1 x_1 + \cdots + \hat{\beta}_j z_j x_j + \cdots + \hat{\beta}_d z_d x_d \\ &= \hat{\alpha}_1 x_1 + \cdots + \hat{\alpha}_j x_j + \cdots + \hat{\alpha}_d x_d, \end{aligned} \tag{3.23}$$

where $(\hat{\alpha}_1, ..., \hat{\alpha}_j, ..., \hat{\alpha}_d)^{\mathsf{T}} = \left(\hat{\beta}_1 z_1, ..., \hat{\beta}_j z_j, ..., \hat{\beta}_d z_d\right)^{\mathsf{T}}$. We may consider $(\hat{\alpha}_1, ..., \hat{\alpha}_j, ..., \hat{\alpha}_d)^{\mathsf{T}}$ as the coefficients of a regression model when the inputs are masked by the feature mask $(z_1, ..., z_j, ..., z_d)^{\mathsf{T}}$. When we set some elements in $(z_1, ..., z_j, ..., z_d)^{\mathsf{T}}$ to 0, the corresponding coefficients in $(\hat{\alpha}_1, ..., \hat{\alpha}_j, ..., \hat{\alpha}_d)^{\mathsf{T}}$ become 0. This strategy leads to sparsity in the regression model, similar to applying $L1$-regularization or elastic-net. In conclusion, our proposed DFR achieves coefficient sparsity. This approach can be seen as a form of regularization for FS because it only selects some of the features.

We then demonstrate that the selector net is necessary because it is designed for computing the feature importance vector $\boldsymbol{c}$. The operator net is neither designed nor trained for this purpose because it could have multiple outputs for multi-class classification. To elaborate, let us consider the three-class scenario shown in Figure 3.8.

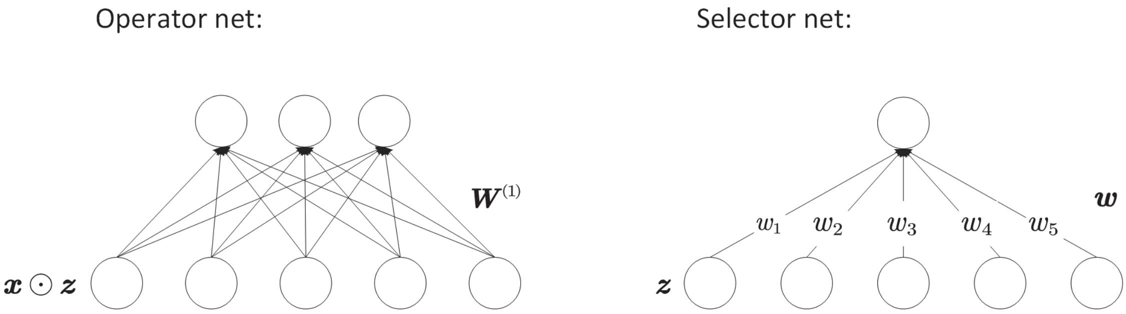


Figure 3.8: A three-class classification task in which the operator net has three output nodes and the selector net only has one output node.

In Figure 3.8, we cannot utilize the weights of the operator net to compute the feature importance vector because the operator net has three output nodes.[8] As a result, the one-to-one correspondence between the input variable $\boldsymbol{x}$ and the weight matrix $\boldsymbol{W}^{(1)}$ of the operator net *cannot* be established.[9] On the other hand, we can compute the feature importance vector using the weights of the selector net, because the selector net has *one* linear output node. We can establish a one-to-one correspondence between the input variable $\boldsymbol{x}$ and the weights of the selector net. In this simple case, the feature importance vector is $\boldsymbol{c} = \boldsymbol{w} = (w_1, w_2, w_3, w_4, w_5)^\mathsf{T}$. In Section 3.2.1, we have explained why $\boldsymbol{c}$ can be derived from a linear regression model. The feature importance vector $\boldsymbol{c}$ can be computed when there is *one* output node. The flexibility of the dual-net architecture enables the selector and operator networks to serve different purposes. Specifically, the one-output selector net computes the importance vector $\boldsymbol{c}$ for FS, and the multi-output operator net performs multi-class

[8] The three-class classification task has three output nodes when the labels are one-hot encoded.

[9] The size of the weight matrix $\boldsymbol{W}^{(1)}$ is 5×3, corresponding to 5 input nodes and 3 output nodes.

classification.

We then demonstrate that our proposed DFR tackles the FS challenge by exploring the correlation between the network weights and the significance of individual input variables. Conventionally, determining the contribution of input variables in a multi-layer neural network is challenging because of the non-linear relationships between the input variables and the network output. Conversely, our proposed DFR is capable of selecting input variables in a multi-layer neural network despite the non-linear relationships. To elaborate this, we begin with a linear regression model, as shown in Figure 3.9.

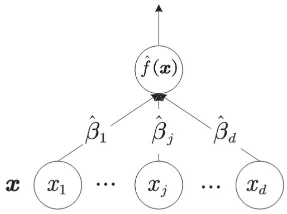


Figure 3.9: A linear regression model in which there is a one-to-one correspondence between input variables $\boldsymbol{x} = (x_1, \cdots, x_j, \cdots, x_d)^\mathsf{T}$ and model parameters $\hat{\boldsymbol{\beta}} = \left(\hat{\beta}_1, \cdots, \hat{\beta}_j, \cdots, \hat{\beta}_d\right)^\mathsf{T}$

Because there is a one-to-one correspondence between the model parameters $\hat{\boldsymbol{\beta}}$ and the input variables $\boldsymbol{x}$, we can interpret the contributions of input variable x_j to the model output through the value of parameter $\hat{\beta}_j$. However, in the multi-layer neural network shown in Figure 3.10(a), the existing FS methods cannot determine the contributions of input variables $\boldsymbol{x}$ to the network output y because $\boldsymbol{x}$ and y are non-linearity related. One method for measuring the contributions of input variables $\boldsymbol{x}$ in a multi-layer neural network is deep feature selection (DFS) [2], which is represented in Figure 3.10(b). DFS utilizes a one-to-one layer, denoted as $\boldsymbol{w}$, connecting $\boldsymbol{x}$ to the first layer of the neural network to measure the contributions of $\boldsymbol{x}$. However, the limitation of this approach is that the one-to-one layer in DFS cannot capture the non-linear relationships between $\boldsymbol{x}$ and y.

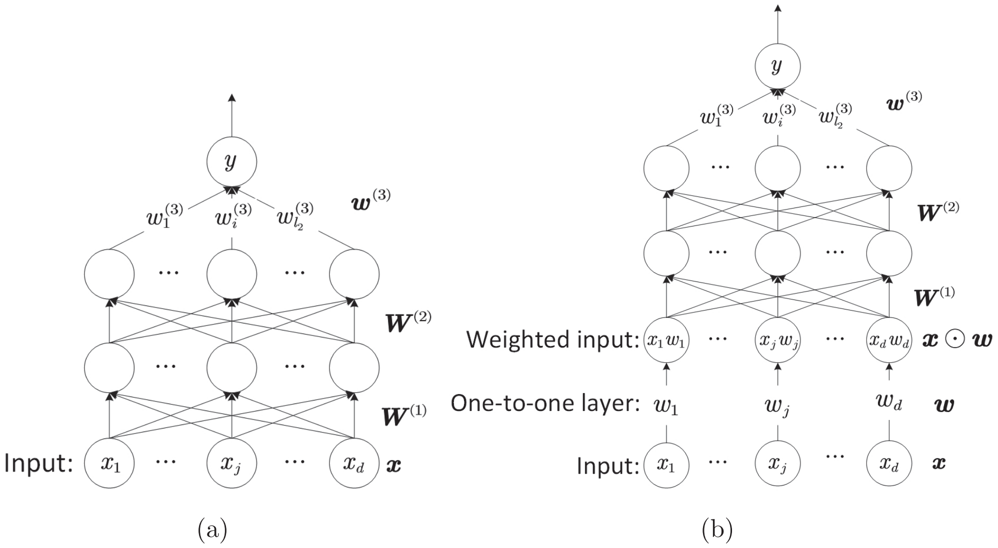


Figure 3.10: (a) A multi-layer neural network with non-linear activations. (b) Deep feature selection (DFS) [2] measures the contributions of $\boldsymbol{x}$ using a one-to-one layer $\boldsymbol{w}$ connecting $\boldsymbol{x}$ to the first layer of the neural network.

To determine the contributions of $\boldsymbol{x}$ to y in Figure 3.10(a), we propose the feature importance vector $\boldsymbol{c} = g\left(g\left(\boldsymbol{W}^{(1)}\right)\boldsymbol{W}^{(2)}\right)\boldsymbol{w}^{(3)} \in \mathbb{R}^d$, where $g(\cdot)$ is a non-linear activation function. Based on the feature importance vector $\boldsymbol{c}$, we can take into account the non-linear relationships between $\boldsymbol{x}$ and y. We can interpret the contributions of $\boldsymbol{x}$ to y through the feature importance vector $\boldsymbol{c}$ and select input variables.

We finally demonstrate that there is a one-to-one correspondence between the input variables $\boldsymbol{x}$ and feature importance vector $\boldsymbol{c}$. We illustrate this correspondence using an example in which the selector net consists of two layers. As proposed before, the one-to-one correspondence of the two-layer network is $g\left(\text{diag}\{\boldsymbol{x}\}\boldsymbol{W}^{(1)}\right)\boldsymbol{w}^{(2)}$, where $\boldsymbol{W}^{(1)}$ is the weight matrix of the hidden layer, $\boldsymbol{w}^{(2)}$ is the weight vector of the output layer, and $g(\cdot)$ is the sigmoid activation function. $g\left(\text{diag}\{\boldsymbol{x}\}\boldsymbol{W}^{(1)}\right)\boldsymbol{w}^{(2)}$ can be expanded as:

$$
\begin{aligned}
& g\left(\operatorname{diag}\{\boldsymbol{x}\}\,\boldsymbol{W}^{(1)}\right)\boldsymbol{w}^{(2)} \\
&= g\left(\begin{bmatrix} x_1 & 0 & \cdots & 0 & \cdots & 0 \\ 0 & x_2 & \cdots & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & x_j & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & \cdots & x_d \end{bmatrix}\begin{bmatrix} w_{1,1}^{(1)} & w_{1,2}^{(1)} & \cdots & w_{1,k}^{(1)} & \cdots & w_{1,l_1}^{(1)} \\ w_{2,1}^{(1)} & w_{2,2}^{(1)} & \cdots & w_{2,k}^{(1)} & \cdots & w_{2,l_1}^{(1)} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ w_{j,1}^{(1)} & w_{j,2}^{(1)} & \cdots & w_{j,k}^{(1)} & \cdots & w_{j,l_1}^{(1)} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ w_{d,1}^{(1)} & w_{d,2}^{(1)} & \cdots & w_{d,k}^{(1)} & \cdots & w_{d,l_1}^{(1)} \end{bmatrix}\right)\begin{bmatrix} w_1^{(2)} \\ w_2^{(2)} \\ \vdots \\ w_k^{(2)} \\ \vdots \\ w_{l_1}^{(2)} \end{bmatrix} \\
&= \begin{bmatrix} g\left(x_1 w_{1,1}^{(1)}\right) & g\left(x_1 w_{1,2}^{(1)}\right) & \cdots & g\left(x_1 w_{1,k}^{(1)}\right) & \cdots & g\left(x_1 w_{1,l_1}^{(1)}\right) \\ g\left(x_2 w_{2,1}^{(1)}\right) & g\left(x_2 w_{2,2}^{(1)}\right) & \cdots & g\left(x_2 w_{2,k}^{(1)}\right) & \cdots & g\left(x_2 w_{2,l_1}^{(1)}\right) \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ g\left(x_j w_{j,1}^{(1)}\right) & g\left(x_j w_{j,2}^{(1)}\right) & \cdots & g\left(x_j w_{j,k}^{(1)}\right) & \cdots & g\left(x_j w_{j,l_1}^{(1)}\right) \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ g\left(x_d w_{d,1}^{(1)}\right) & g\left(x_d w_{d,2}^{(1)}\right) & \cdots & g\left(x_d w_{d,k}^{(1)}\right) & \cdots & g\left(x_d w_{d,l_1}^{(1)}\right) \end{bmatrix}\begin{bmatrix} w_1^{(2)} \\ w_2^{(2)} \\ \vdots \\ w_k^{(2)} \\ \vdots \\ w_{l_1}^{(2)} \end{bmatrix} \\
&= \begin{bmatrix} \sum_{k=1}^{l_1} g\left(x_1 w_{1,k}^{(1)}\right) w_k^{(2)} \\ \sum_{k=1}^{l_1} g\left(x_2 w_{2,k}^{(1)}\right) w_k^{(2)} \\ \vdots \\ \sum_{k=1}^{l_1} g\left(x_j w_{j,k}^{(1)}\right) w_k^{(2)} \\ \vdots \\ \sum_{k=1}^{l_1} g\left(x_d w_{d,k}^{(1)}\right) w_k^{(2)} \end{bmatrix} = \begin{bmatrix} v_1(x_1) \\ v_2(x_2) \\ \vdots \\ v_j(x_j) \\ \vdots \\ v_d(x_d) \end{bmatrix}.
\end{aligned}
\tag{3.24}
$$

Eq. (3.24) shows that there is a one-to-one correspondence between v_j and x_j.

Chapter 4

EXPERIMENTS ON DUAL DROPOUT RANKING (DDR)

This chapter details the experimental setup and results on dual dropout ranking (DDR). The DDR is evaluated on the ADReSS and AD2021 datasets.

4.1 Datasets

4.1.1 The ADReSS Dataset

The AD Recognition Through Spontaneous Speech Challenge (ADReSS) [74] provides a benchmark dataset and a platform where the research community can compare their methods for improving AD detection performance. The dataset comprises recordings of the spoken-language descriptions of the Cookie Theft picture description task in Boston Diagnostic Aphasia Examinations. 156 subjects aged between 50 to 80 participated in the examinations, among whom 78 are AD patients and 78 are healthy control (HC). Among these participants, 108 were grouped into the training set, and the remaining 48 were grouped into the test set. The dataset is gender-balanced, and the spoken language is English. Table 4.1 shows the dataset's details.

4.1.2 The AD2021 Dataset

The AD2021 dataset [75] was released through an AD recognition competition organized by Jiangsu Normal University, SATLab of Tsinghua University, and Beijing Haitian Ruisheng Science Technology Ltd. The dataset comprises the speech recordings of "Cookie Theft picture description" sessions, fluency tests, and normal conversations. The training set contains 25 AD patients, 53 older adults suffering from MCI, and 44 HC. Each subject in the training set has several recording sessions, resulting in 279 training sessions. The test set contains 119 subjects, of which 35 are AD patients, 39 have MCI, and 45 are HC. The spoken language of the dataset is Mandarin Chinese. No manual transcription is provided. Table 4.1 shows the dataset's details.

Table 4.1: The characteristics of the ADReSS and the AD2021 datasets. *AD*: Alzheimer's disease, *MCI*: mild cognitive impairment, *HC*: healthy control, *M*: male, and *F*: female.

Dataset		**ADReSS**								**AD2021**											
Training/test data		Training data				Test data				Training data						Test data					
Class		HC		AD		HC		AD		HC		MCI		AD		HC		MCI		AD	
Gender		M	F	M	F	M	F	M	F	M	F	M	F	M	F	M	F	M	F	M	F
Age	[50, 55)	1	0	1	0	1	0	1	0	18	26	27	27	10	15	22	23	10	29	6	29
	[55, 60)	5	4	5	4	2	2	2	2												
	[60, 65)	3	6	3	6	1	3	1	3												
	[65, 70)	6	10	6	10	3	4	3	4												
	[70, 75)	6	8	6	8	3	3	3	3												
	[75, 80)	3	2	3	2	1	1	1	1												
Number of samples		54		54		24		24		108		93		78		45		39		35	
Spoken language		English								Mandarin Chinese											
Task		Cookie theft picture description								Cookie theft picture description, fluency test, and normal conversation*											
Manual transcriptions provided		Yes								No											

*Our experiments adhered to the official guideline by utilizing all the three tasks.

4.2 *Feature Engineering*

We focus on two categories of features: transcription-based and speech-based. The transcription-based features are extracted from either the manual or automatic transcriptions, which capture the semantic, syntactic, and lexical aspects of the speaker's spoken language. The speech-based features contain a variety of acoustic characteristics of the speakers. For the ADReSS dataset [74], the transcription-based features significantly outperform the speech-based features [74, 76] because accurate manual transcriptions are provided; therefore, we focus on the transcription-based features only. For the AD2021 dataset [75], because of erroneous transcriptions, we include various types of speech-based features in addition to the transcription-based features.

4.2.1 Features for the ADReSS Dataset

Linguistic Features

34 linguistic features were extracted from the CHAT annotated transcriptions using the EVAL command in the CLAN program [74]. The features include lengths of utterances, type-token ratios, statistics of POS, etc.[1] The ADReSS challenge [74] has provided a baseline recognition performance on the linguistic features.

BERT Features

The BERT model [26], which comprises deep bidirectional Transformers, has been widely adopted in natural language processing (NLP). A pre-trained BERT model can be fine-tuned to suit a wide range of tasks. In [32], the authors fine-tuned a BERT model at the transcription-level for AD recognition and achieved impressive results. In this paper, we use the pre-trained BERT model as a *feature extractor*. More specifically, we fed the subjects' transcriptions to the pre-trained BERT model and extracted the representations from the last layer of the model. For each subject, the model produces a 768-dimensional feature vector (called the BERT features) that abstractly captures the semantic, syntactic, and lexical information of the transcriptions. Li *et al.* [76] extracted BERT features from both manual and automatic transcriptions. Their results demonstrate the effectiveness of the BERT features for dementia detection.

[1] Linguistic features are listed in Appendix C.

Pause Features

In [77], the authors demonstrated that pauses can function as word-finding, as planning at the word, phrase, and narrative levels, and as pragmatic compensation when other interactional and narrative skills deteriorate. In [32], pause information was incorporated into the feature representations to improve AD recognition performance. Thus, we included the pause features for dementia detection.

We used the pause statistics in Table 4.2 as the pause features. To obtain these features, we followed the procedure in [32] and used the 'chat2text' command in CLAN to convert the CHAT annotated transcriptions into plain words and tokens. Then, the converted transcriptions were forced aligned with the speech recordings using the Penn Phonetics Lab Forced Aligner [78]. The outputs of the alignments contain the identifications and durations of the pauses.[2]

We divided the pauses into six duration groups: G_1 (pauses between 0.05s–0.5s), G_2 (pauses between 0.5s–1s), G_3 (pauses between 1s–2s), G_4 (pauses between 2s–3s), G_5 (pauses between 3s–4s), and G_6 (pauses longer than 4s). For each duration group, we extracted the five pause features in Table 4.2. As a result, we had a total of $5 \times 6 = 30$ pause features per recording.

Table 4.2: Five pause features extracted from the six duration groups (listed in Section 4.2.1).

Pause feature	**Description**
#p	Number of pauses per minute
%p/word ratio	Pause-to-word ratio
p duration	Total duration of pauses per minute
p mean duration	Mean duration of pauses
%p duration/word duration	Pause-duration-to-word-duration ratio

[2] The between-word pauses are indicated by 'sp'.

4.2.2 Features for the AD2021 Dataset

Lexical Features

Because manual transcriptions are not available in AD2021, ASR was used before extracting the lexical features. The Tencent Cloud ASR[3] was adopted to transcribe the Mandarin speech recordings. Based on the transcriptions, the following lexical features were extracted:[4] the number of sentences per minute, the average number of words per sentence, the ratio of unique words to all words, and the average word frequency. Then, the Stanford POS tagger[5] was utilized to parse the transcriptions to extract the following lexical features: POS counts per minute, POS ratio, the ratio of pronoun to noun, the ratio of noun to verb, the maximum parsed tree height, the mean parsed tree height, and the median parsed tree height. These lexical features lead to a 143-dimensional feature vector per recording.[6]

BERT Features

Similar to the ADReSS dataset, a pre-trained Chinese BERT model[7] was employed as the feature extractor. The transcriptions were fed to the BERT model and high-level representations were extracted from the last layer of the model, resulting in a 768-dimensional vector per recording.

[3] https://cloud.tencent.com/product/asr

[4] The lexical features are extracted using this toolbox: https://github.com/SPOClab-ca/COVFEFE.

[5] https://nlp.stanford.edu/software/tagger.shtml

[6] Lexical features are listed in Appendix C.

[7] https://huggingface.co/bert-base-chinese

Acoustic Features

We followed the standard pipelines in the COVFEFE toolbox[8] to extract the acoustic features from the speech recordings, which include formants, loudness, pitch, zero-crossing rate, etc.

INTERSPEECH 2010 Paralinguistic Challenge Features (IS10)

IS10 [79] is a feature set for emotion recognition and bipolar disorder recognition. In addition to the 32 low-level descriptors (LLDs) in INTERSPEECH 2009 Emotion Challenge (IS09), 44 LLDs were added to IS10, including PCM loudness, eight log Mel-frequency bands, eight line-spectral frequency pairs, fundamental frequency (F0) envelope, voicing probability, jitter, and shimmer. Twelve statistics (minimum, maximum, mean, range, etc.) of the LLDs were computed, leading to a 1582-dimensional feature vector per recording.

COVAREP Features

COVAREP [41] provides comprehensive acoustic features, which include prosodic features (F0 and voicing), voice quality features, and spectral features. We extracted COVAREP features at 100Hz; for each recording, the mean, maximum, minimum, median, standard deviation, skew, and kurtosis of the features were computed, leading to a 518-dimensional feature vector per recording.

Pause Features

An energy-based voice activity detector (VAD) was utilized to identify the pauses. Similar to the ADReSS dataset, the pauses were divided into six groups, and pause features (Table 4.2) were determined from individual groups, leading to a 30-dimensional feature vector per recording.

[8] https://github.com/SPOClab-ca/COVFEFE

4.2.3 Implementation Details of DDR

Both the operator network and selector network in DDR are feedforward neural networks. A batch-normalization layer followed by a dropout layer with a dropout rate of 0.5 was added after each hidden layer in the two networks. The activation function for the hidden layers is ReLU for both networks, while the activation function for the last layer of the operator network is softmax and that for the selector network is linear. An Adam optimizer with a learning rate of 0.001 was used to optimize the networks' parameters and the trainable dropout rates, which were initialized to 0.35. The batch size $|\mathcal{M}|$ was set to 32 and the size of the dropout mask subset $|\mathcal{Z}|$ was set to 32. On a Ubuntu 20.4 machine with one RTX3090 GPU, each experiment took about 5 minutes.

4.2.4 Performance Metrics

The goal is to determine the most discriminative features that can effectively identify individuals who are HC, those with MCI and others with AD. The two-step FS method described in Section 3.1.4 was utilized to identify the discriminative features. For the ADReSS dataset, the identified features were then used for training linear SVM classifiers[9] with a box constraint of 1 to classify AD and HC. For the AD2021 dataset, the selected features were used for training Gaussian SVM classifiers[10] with a box constraint of 1 to identify AD, MCI, and HC.

The performance metrics for the ADReSS dataset include precision (PRE), recall (REC), and F_1 scores for each class (AD and HC) as well as their unweighted mean and accuracy. The performance on the training set was obtained by 10-fold CV.

For the AD2021 dataset, except for the accuracy, the performance metrics were calculated for each class (AD, MCI, and HC) and their unweighted mean was reported.

[9] The classifier's setting was adopted from [76]. The SVMs were forced to produce probabilistic outputs when computing the predicted scores.

[10] The classifier's setting was taken from the AD2021 competition baseline.

The 10-fold CV was replaced by a leave-n-subject-out CV in which the training samples of the same speakers were grouped into either the TR or the TS for each fold.

4.3 Experiments and Results

In this section, we first evaluate DDR on a synthetic dataset and the MNIST handwritten digit dataset and then evaluate the two-step FS approach on the ADReSS and AD2021 datasets.

4.3.1 Analysis of Keep Probabilities on a Synthetic Dataset

A synthetic data set was designed to evaluate the capability of classifiers and FS algorithms in solving a multi-dimensional XOR problem [80]. By grouping the eight corners of a 3-dimensional hypercube $(v_0, v_1, v_2) \in \{-1, 1\}^3$ into the tuples (v_0v_2, v_1v_2), we have 4 sets of vectors and their negations $\{\boldsymbol{v}^{(c)}, -\boldsymbol{v}^{(c)}\}_{c=1}^4$, where c is the class index. For example, the tuple $(v_0v_2, v_1v_2) = (-1, -1)$ corresponds to $c = 2$, where $\boldsymbol{v}^{(2)} = [1, 1, -1]^\mathsf{T}$. The points in class c are generated from the distribution $\frac{1}{2}[\mathcal{N}(\boldsymbol{v}^{(c)}, 0.5\boldsymbol{I}_3) + \mathcal{N}(-\boldsymbol{v}^{(c)}, 0.5\boldsymbol{I}_3)]$, where $\boldsymbol{I}_3$ is a 3×3 identity matrix and $\mathcal{N}(\mu, \sigma)$ is a Gaussian distribution. Each sample is additionally accompanied by 7 Gaussian noise features with zero mean and unit variance, leading to a 10-dimensional feature vector.

We trained a dual network (Figure 3.2) on the synthetic data for FR. After training, the keep probabilities $(\boldsymbol{1} - \boldsymbol{\theta})$ of the features for 20 random seeds are depicted in Figure 4.1. It shows that the keep probabilities associated with the valid features (v_0, v_1, v_2) converge to 1, whereas the noise features $(v_3 \sim v_9)$ have keep probabilities close to 0. This result suggests that DDR can effectively identify the valid features.

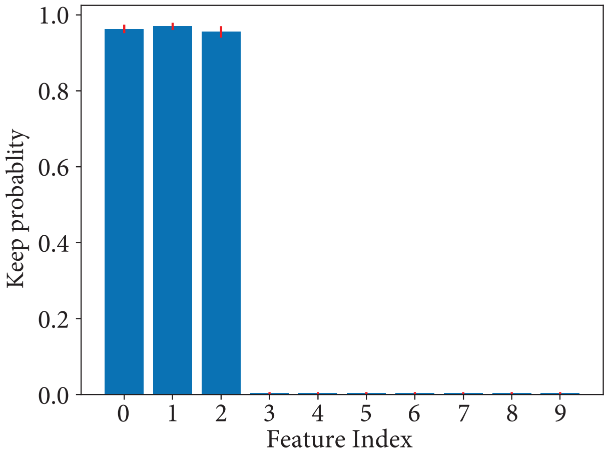


Figure 4.1: The keep probabilities $(\mathbf{1} - \boldsymbol{\theta})$ of 10 features in the synthetic dataset for 20 random seeds. Indexes 0–2 and 3–9 correspond to the valid and invalid features, respectively. The blue bars and the red error bars denote the means and two times the standard deviations of 20 random seeds, respectively.

4.3.2 Visualizing the Keep Probabilities

To further demonstrate the explainability of DDR, we employed the MNIST handwritten digit dataset for binary classification. More specifically, we utilized a subset of the MNIST dataset to distinguish digits '3' and '8'. We flattened the 28×28 digits into 784-dimensional feature vectors as inputs. After training, we normalized and reshaped the keep probabilities $(\mathbf{1} - \boldsymbol{\theta})$ into a 28×28 matrix to represent the feature importance map. We applied 5-fold CV for evaluation. For each fold, we trained a dual-net and selected 50 features. The selected features were then used to train a Gaussian SVM with a box constraint of 1 to classify digits '3' and '8'. We achieved an accuracy of 0.981 ± 0.003 based on the selected features.[11] The feature importance map is shown in Figure 4.2. It shows that DDR can identify the relevant features despite the flattening process destroying the images' spatial information.

[11] The codes are available at `https://github.com/kexquan/dual-dropout-ranking`.

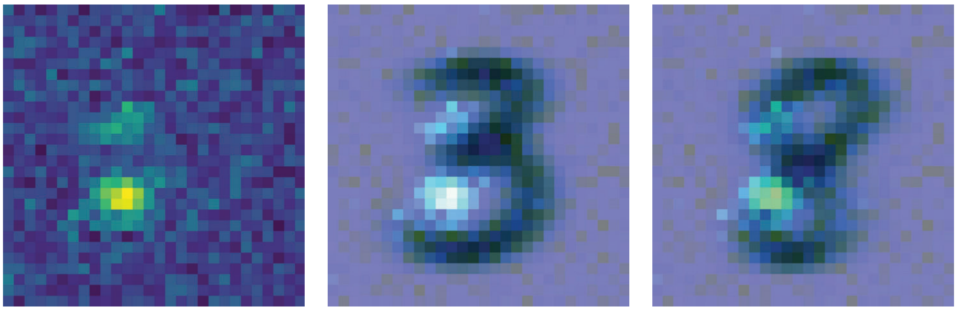


Figure 4.2: A Feature importance map produced by a selector trained on MNIST data. The left picture is the normalized feature importance map. The middle and the right pictures are the feature importance map superimposed on the mean images of digit '3' and digit '8', respectively.

4.3.3 Performance of Different Feature Types

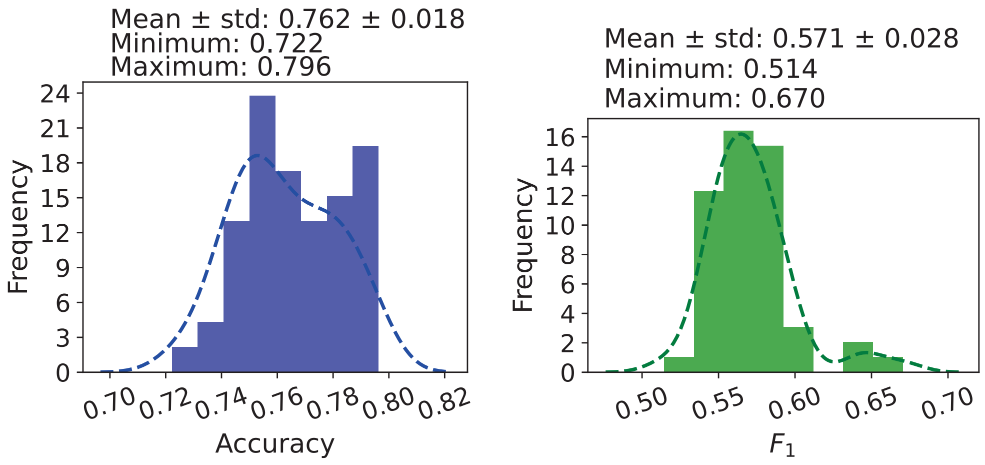
We first evaluated the recognition performance of all the feature sets *before* FS. We ran 100 repetitions of 10-fold CV and averaged the performance values. The corresponding results are reported in Table 4.3. The results show that on the ADReSS training data, the linguistic features achieve the best performance before FS. On the AD2021 training data, the IS10 feature set achieves the best performance among all the feature sets. The transcription-based features (lexical and BERT) perform worse than the speech-based features. This may be due to word errors in the automatic transcriptions.

Table 4.3: Classification performance on the ADReSS and the AD2021 training data before FS. The numbers in the brackets are the sizes of the feature sets. *ACC*: accuracy; *PRE*: precision; *REC*: recall.

Dataset	**Feature set**	**10-fold CV on training data**			
		ACC	PRE	REC	F_1
ADReSS	Linguistic (34)	***0.802***	***0.806***	***0.799***	***0.783***
	BERT (768)	0.748	0.737	0.776	0.735
	Pause (30)	0.523	0.534	0.446	0.454
AD2021	Lexical (143)	0.553	0.479	0.511	0.450
	BERT (768)	0.575	0.514	0.530	0.482
	Acoustic (30)	0.613	0.575	0.565	0.519
	COVAREP (518)	***0.678***	0.636	0.628	0.578
	IS10 (1582)	0.666	***0.638***	***0.642***	***0.587***
	Pause (30)	0.351	0.308	0.324	0.281

4.3.4 Performance of Filter Methods

For the ADReSS dataset, we combined all the features to form 832-dimensional vectors. The dimensionality of the combined features for the AD2021 training data is 3071. When conducting 10-fold CV on the combined features, large differences in recognition performance across CV were observed, as illustrated in Figure 4.3.



(a)

(b)

Figure 4.3: When conducting 10-fold CV based on different data splittings, large variations in recognition performance across CV were observed on (a) the ADReSS and (b) the AD2021 training data.

This is because during the CV, applying random splitting on a limited number of training samples will induce great differences across TR in different folds. These large differences suggest recognition performance on unseen data is likely to be brittle. To mitigate this brittleness, we propose the following ensemble procedure to stabilize the classification performance during CV. We ran I repetitions of CV based on different data splittings. We then produced the predicted scores $p(i, j)$ for subject j in CV i. Finally, we averaged the predicted scores $p(j) = (1/I) \sum_{i=1}^{I} p(i, j)$ over all the CV for each of the J subjects, as shown in Figure 4.4.

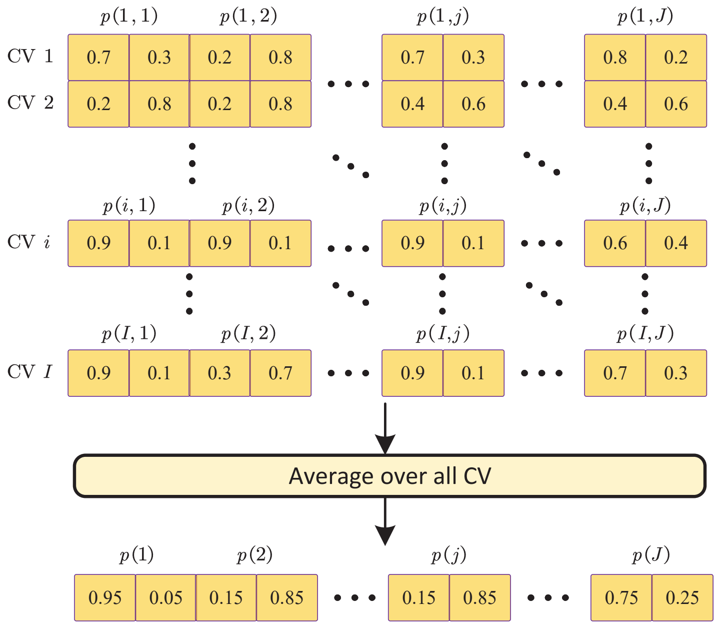


Figure 4.4: The ensemble procedure to stabilize the classification performance during CV. We ran I repetitions of CV based on different data splittings and averaged the predicted scores $p(i, j)$ over all the CV for each of the J subjects.

To test our proposed ensemble procedure, we ran 50 repetitions of CV based on different data splittings. From the 50 CV, we selected five CV ($m = 5$) and averaged the predicted scores over the five CV. The results in the first row of Table 4.4 summarize 100 draws of the five CV. The second row is similar, except $m = 10$. Comparing the legend of Figure 4.3 and Table 4.4, we can see that the ensemble procedure increases the mean accuracy and F_1 and reduces variances on both datasets. On the ADReSS training data, when $m = 25$, the ensemble procedure achieves the highest mean accuracy and boosts the minimum accuracy from 0.722 (Figure 4.3(a)) to 0.750. On the AD2021 training data, the ensemble procedure achieves the highest mean F_1 and boosts the minimum F_1 from 0.514 (Figure 4.3(b)) to 0.555 when $m = 10$. Therefore, subsequent experiments repeated the CV 25 times and averaged the predicted scores on the ADReSS dataset. On the AD2021 dataset, we conducted 10 repetitions of CV and averaged the predicted scores.

Table 4.4: The proposed ensemble procedure improves mean classification performance and reduce variances. m is the ensemble size. ACC: accuracy.

	ADReSS (ACC)		AD2021 (F_1)	
m	Mean ± std	Min – Max	Mean ± std	Min – Max
5	0.768 ± 0.013	0.741 – 0.796	0.571 ± 0.010	0.552 – 0.603
10	0.769 ± 0.010	0.741 – 0.787	$\boldsymbol{0.573 \pm 0.007}$	0.555 – 0.588
15	0.771 ± 0.009	0.750 – 0.787	0.570 ± 0.007	0.552 – 0.588
20	0.773 ± 0.009	0.750 – 0.787	0.571 ± 0.006	0.561 – 0.582
25	$\boldsymbol{0.773 \pm 0.008}$	0.750 – 0.787	0.570 ± 0.005	0.555 – 0.582
30	0.771 ± 0.008	0.759 – 0.787	0.571 ± 0.006	0.555 – 0.585
35	0.771 ± 0.008	0.759 – 0.787	0.572 ± 0.005	0.561 – 0.585
40	0.771 ± 0.008	0.759 – 0.787	0.571 ± 0.005	0.558 – 0.582
45	0.771 ± 0.006	0.759 – 0.787	0.571 ± 0.004	0.564 – 0.582

We followed the procedure described in Section 3.1.4 to evaluate the classification performance of the filter methods (FDR, PeaCorr, and MutInfo) on the combined feature vectors. Note that FS was performed *inside* the CV, and each fold may select different features because the TR in Figure 3.3 were different for different folds. On the TR of individual folds, we applied the filter methods to reduce the feature dimension to $n = \{25, 50, 100, 150, \ldots, 600\}$, as shown in Table 4.5. It shows that using the filter methods to pre-screen the combined features can improve classification performance on both datasets. On the ADReSS training data, MutInfo achieves the highest accuracy (0.796) when the feature dimension was reduced to 50. On the AD2021 training data, MutInfo achieves the highest F_1 scores (0.641) when the feature dimension was reduced to 100. Therefore, in the two-step FS, subsequent experiments utilized MutInfo to pre-screen the combined feature vectors to 50 and 100 for ADReSS and AD2021, respectively.

Table 4.5: Classification performance of the filter methods on the ADReSS and the AD2021 training data. n: the number of selected features. ACC: accuracy.

	ADReSS (ACC)			AD2021 (F_1)		
n	FDR	PeaCorr	MutInfo	FDR	PeaCorr	MutInfo
25	0.741	0.741	0.778	0.559	0.596	0.623
50	0.759	0.769	***0.796***	0.567	0.592	0.640
100	0.769	0.769	0.759	0.586	0.588	***0.641***
150	0.731	0.741	0.787	0.568	0.588	0.623
200	0.741	0.741	0.787	0.568	0.604	0.601
250	0.778	0.778	0.778	0.585	0.601	0.597
300	0.778	0.778	0.787	0.586	0.592	0.602
350	0.787	0.787	0.778	0.582	0.595	0.594
400	0.778	0.778	0.787	0.594	0.595	0.583
450	0.778	0.778	0.787	0.591	0.595	0.586
500	0.787	0.787	0.769	0.585	0.592	0.585
550	0.787	0.787	0.769	0.582	0.598	0.588
600	0.796	0.796	0.759	0.577	0.588	0.579

4.3.5 Performance of Two-step FS on Training Data

This subsection reports the performance of DDR and some strong supervised FS methods on the ADReSS and the AD2021 training data. These strong supervised FS methods include DFS [2], DropoutFR [4], and FIR [3]. On the TR of individual folds, after using MutInfo to pre-screen the combined features, we applied DDR and these strong supervised FS methods on the remaining 50 features for ADReSS and 100 features for AD2021 to further select relevant features. We adopted the same network architectures (“50–128–32–2” for ADReSS and “100–128–128–32–3” for AD2021) with softmax outputs and default hyper-parameters in the source codes for these strong supervised FS methods and DDR. During the CV, we selected the same number of features n in each fold for each of the FS methods. The results on the ADReSS training data are shown in Table 4.6, and results on the AD2021 training data are shown in Table 4.7. The results show that applying DDR and these strong supervised FS methods on the pre-screened features can further improve recognition performance. The two-step FS significantly reduces feature dimensionality while identifying small feature subsets that achieve comparable or superior performance compared with the combined feature sets. The results also show that DDR performs the best on both datasets, that is, it achieves the best mean recognition performance among these FS methods.

Table 4.6: Recognition performance of the two-step FS on the ADReSS training data. Features were pre-screened by MutInfo. n: the number of selected features in each fold. ACC: accuracy.

	CV on training data (ACC)			
n	DFS [2]	DropoutFR [4]	FIR [3]	DDR (Ours)
5	0.778	0.769	0.778	0.815
10	0.787	0.796	0.778	0.815
15	0.787	0.787	0.778	0.787
20	0.787	0.787	0.769	0.796
25	0.787	0.806	0.769	0.787
Mean	0.785	0.789	0.774	***0.800***

Table 4.7: Recognition performance of the two-step FS on the AD2021 training data. Features were pre-screened by MutInfo. n: the number of selected features in each fold.

	CV on training data (F_1)			
n	DFS [2]	DropoutFR [4]	FIR [3]	DDR (Ours)
5	0.705	0.663	0.652	0.744
10	0.738	0.714	0.772	0.734
15	0.774	0.736	0.763	0.752
20	0.763	0.744	0.777	0.751
25	0.726	0.760	0.767	0.757
30	0.731	0.760	0.760	0.773
35	0.718	0.729	0.748	0.742
40	0.691	0.701	0.719	0.727
45	0.686	0.692	0.698	0.699
50	0.664	0.679	0.670	0.689
Mean	0.720	0.718	0.733	***0.737***

4.3.6 Performance of Two-step FS on Test Data

This subsection reports the performance of the identified feature subsets on the ADReSS and AD2021 test data. During CV, each fold may select different feature subsets because the TR in Figure 3.3 are different for different folds. When evaluating the selected feature subsets on test data, we utilized the following *soft voting* procedure to incorporate these different feature subsets. We utilized SVM classifiers to produce the predicted scores $p(k)$ for the k-th feature subset. We then averaged the predicted scores $p = (1/K) \sum_{k=1}^{K} p(k)$ over all the K feature subsets for the final classification. We computed the results of different sizes of feature subsets and averaged the results in Table 4.8. We also compared our methods with some recent results in Table 4.8. On the AD2021 test data, “MutInfo + DDR” achieves the highest recognition performance among all the methods. On the ADReSS test data, the proposed two-step FS significantly performs better than the official baseline. “MutInfo + DDR” also outperforms the best reported results in the ADReSS challenge [32]. Additionally, Table 4.8 supports the following key findings:

1) Our method performs FS on the combined feature vectors (official baseline features [74] + pause features + BERT features [76]). On this basis, our method not only reduces feature dimension but also boosts the accuracy of the official baseline [74] from 75% to 90%.

2) Compared to using the BERT features [76] only, our method can select features that increase the accuracy from 87.5% to 90.4%, while the features selected by“MutInfo + DFS” [2] reduce the accuracy from 87.5% to 86.3%.

3) Our method yields superior performance to “MutInfo + DropoutFR” [4]. Specifically, while the features selected by the latter increase the accuracy from 87.5% to 89.6%, the accuracy achieved by our method is even higher (90.4%).

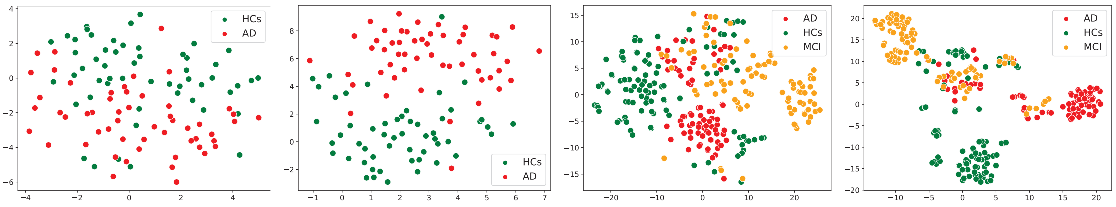
4) While “MutInfo + FIR” [3] improves the accuracy from 87.5% to 90.0%, it reduces the REC for the AD class from 83.3% to 80.0%. As a result, “MutInfo + FIR” diagnoses fewer AD patients than using the BERT features alone. In contrast, our method not only improves the accuracy to 90.4% but also maintains the REC for the AD class.

Table 4.8: Recognition performance of the two-step FS on the ADReSS and AD2021 test data. *ACC*: accuracy; *PRE*: precision; *REC*: recall.

Dataset	**Method**	Class/mean	**Performance on test data**			
			PRE	REC	F_1	ACC
ADReSS	Official baseline (Linguistic) [74]	HC	0.700	0.870	0.780	
		AD	0.830	0.620	0.710	0.750
		Mean	0.765	0.745	0.745	
	Pause	HC	0.680	0.708	0.694	
		AD	0.696	0.667	0.681	0.688
		Mean	0.688	0.688	0.687	
	BERT [76]	HC	0.846	0.917	0.880	
		AD	0.909	0.833	0.870	0.875
		Mean	0.878	0.875	0.875	
	Text modality + label fusion [38]	Mean	–	–	–	0.854
	ERNIE3p [32]	Mean	–	–	–	0.896
	BERT + ViT [46]	Mean	0.871	0.892	0.880	0.879
	MutInfo + DFS [2]	HC	0.796	0.975	0.876	
		AD	0.968	0.750	0.845	0.863
		Mean	0.882	0.863	0.861	
	MutInfo + DropoutFR [4]	HC	0.852	0.958	0.902	
		AD	0.952	0.833	0.889	0.896
		Mean	0.902	0.896	0.895	
	MutInfo + FIR [3]	HC	0.833	1.000	0.909	
		AD	1.000	0.800	0.889	0.900
		Mean	***0.917***	0.900	0.899	
	MutInfo + DDR (Ours)	HC	0.855	0.975	0.911	
		AD	0.972	0.833	0.897	***0.904***
		Mean	0.913	***0.904***	***0.904***	
AD2021	Official baseline (IS10)[12]	Mean	0.799	0.785	0.786	0.798
	Lexical[13]	Mean	0.738	0.602	0.578	0.630
	Pause	Mean	0.422	0.425	0.421	0.437
	Acoustic[14]	Mean	0.651	0.648	0.647	0.655
	COVAREP [41]	Mean	0.717	0.703	0.704	0.706
	BERT[15]	Mean	0.674	0.620	0.615	0.639
	Wav2vec 2.0 [75]	Mean	0.830	0.828	0.828	0.832
	Adversarial self-supervised model [81]	Mean	0.838	0.837	0.837	–
	MutInfo + DFS [2]	Mean	0.858	0.852	0.851	0.852
	MutInfo + DropoutFR [4]	Mean	0.864	0.861	0.860	0.862
	MutInfo + FIR [3]	Mean	0.862	0.855	0.854	0.857
	MutInfo + DDR (Ours)	Mean	***0.875***	***0.869***	***0.867***	***0.871***

4.3.7 Analysis of Selected Features

Figure 4.5 depicts the t-SNE plots of the ADReSS and AD2021 training data. Figure 4.5(b) shows that the selected features distinguish the two groups with a bigger gap. Figure 4.5(d) shows that the selected features reduce the intra-group distance, although there is still some overlap between the groups.



(a) (b) (c) (d)

Figure 4.5: 2D t-SNE plots of the ADReSS training data based on (a) all feature sets and (b) 30 features selected by "MutInfo + DDR" with the highest selection frequency. The selected features distinguish the two groups with a bigger gap. 2D t-SNE plots of the AD2021 training data based on (c) all feature sets and (d) 30 features selected by "MutInfo + DDR" with the highest selection frequency. The selected features reduce the intra-group distance.

We then depict 100 features selected by "MutInfo + DDR" with the highest selection frequency in Figure 4.6. Figure 4.6(a) shows that although none of the features were selected in all folds, among the 1250 folds, 1083 folds selected the most common feature. The most commonly selected features are BERT features. Additionally, two of the pause features and some of the linguistic features were selected, as shown in Table 4.9. Figure 4.6(b) shows that in the AD2021 dataset, still no feature was se-

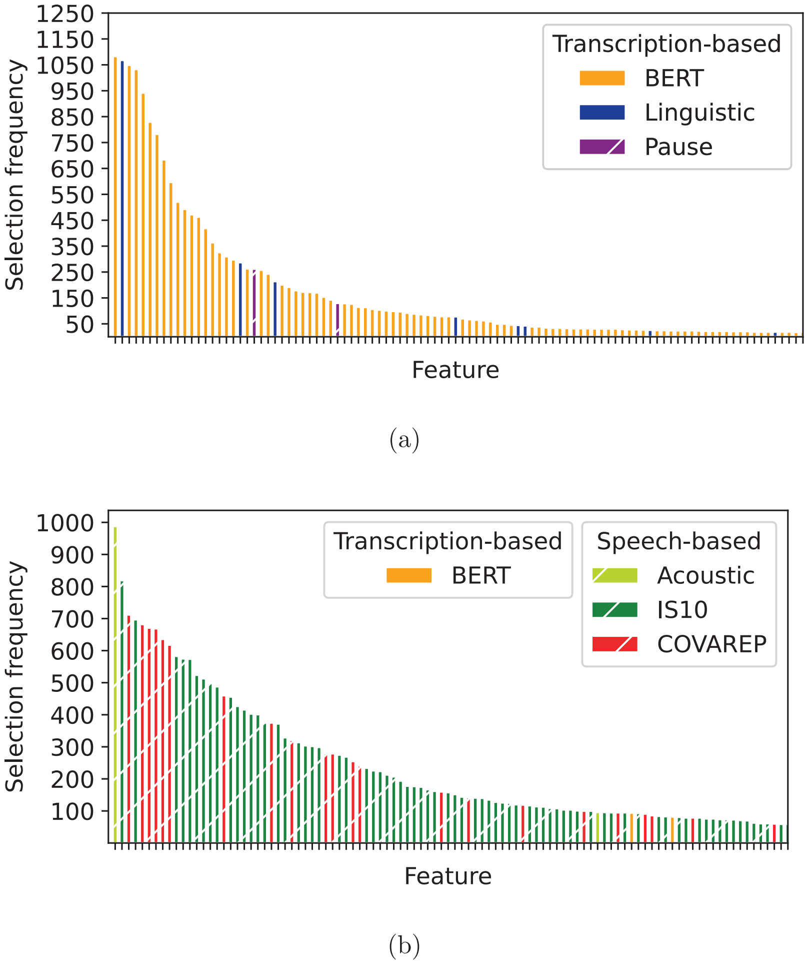
[12] https://github.com/THUsatlab/AD2021

[13] The lexical features are extracted using this toolbox: https://github.com/SPOClab-ca/COVFEFE.

[14] The acoustic features are extracted using this toolbox: https://github.com/SPOClab-ca/COVFEFE.

[15] https://huggingface.co/bert-base-chinese

lected in all 1000 folds, but the most common one appears in 988 folds. COVAREP and IS10 features were the most commonly selected features. This is reasonable because COVAREP and IS10 features perform well on the training data. Only a few transcription-based features were selected. This may be due to the transcription errors. Compared with the ADReSS dataset, the performance of transcription-based features in AD2021 is unsatisfactory. None of the pause features rank above the top 100.



(a)

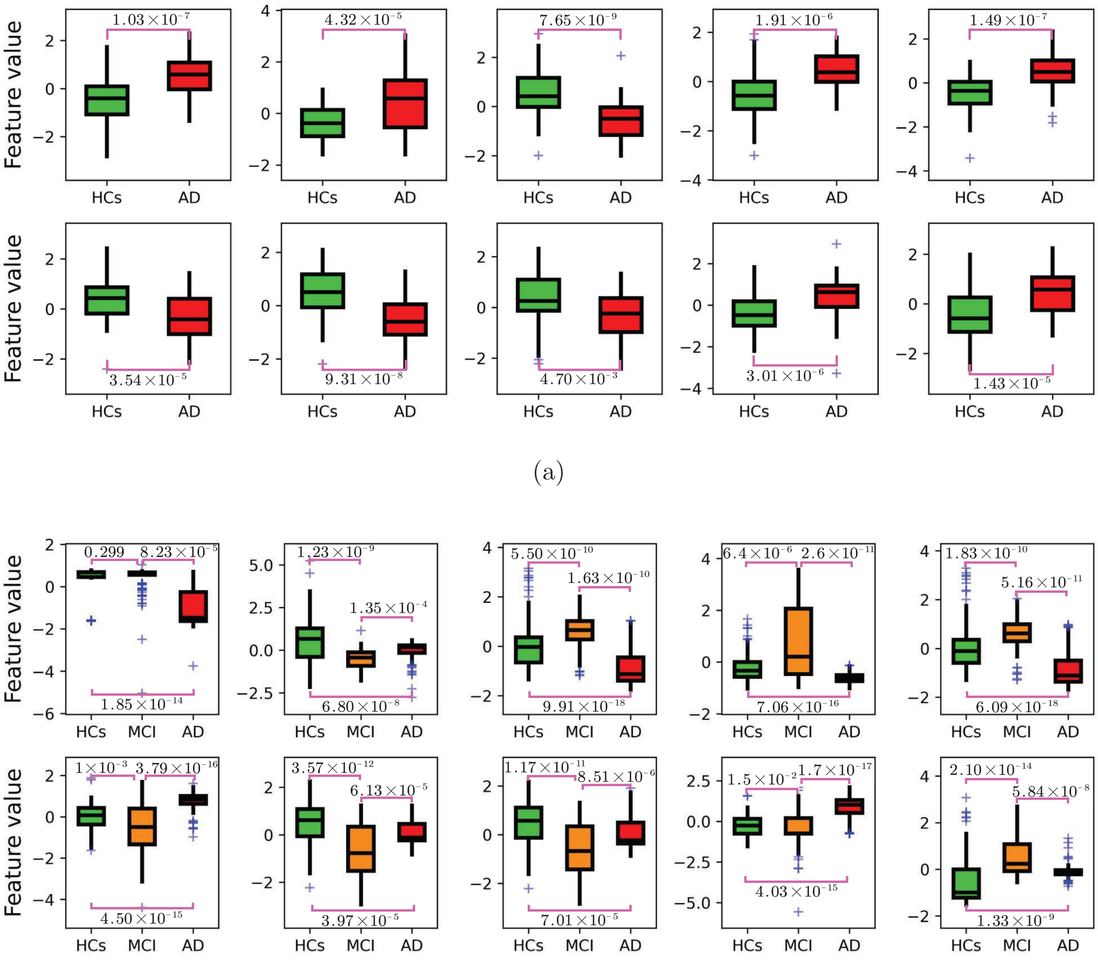
(b)

Figure 4.6: 100 features selected by "MutInfo + DDR" with the highest selection frequency on (a) the ADReSS and (b) the AD2021 training data.

Table 4.9: The linguistic features and pause features discovered by "MutInfo + DDR" on the ADReSS training data. The parenthesized values are the frequency of the features being selected during the CV. *AD*: Alzheimer's disease, *HC*: Healthy control.

Feature	**Known specificity**
% pro: Percentage of pronouns (1068)	Ahmed *et al.* [82] reported changes in *the number of pronouns*, and Jarrold *et al.* [83] reported an increase in *the proportion of pronouns* in AD patients.
% Nouns: Percentage of nouns (287)	Jarrold *et al.* [83] reported a decrease in *the proportion of nouns* in AD patients.
%p/word ratio: (Pauses between 0.05s–0.5s)-to-word ratio (262)	–
Words/min: Words per minute (214)	AD could be detected through the analysis of voice activity detection and *speech rate* tracking [84].
%p duration/word duration: (pauses between 2s–3s)-duration-to-word-duration ratio (130)	–
noun/verb ratio: Total no. of nouns / total no. of verbs (78)	AD patients may have more difficulty *naming verbs than nouns* [39], and Robinson *et al.* [85] found that AD patients performed worse on a picture-naming task for *verbs than nouns*.

We finally depict the box plots of top 10 selected features in Figure 4.7. Figure 4.7(a) shows that on the ADReSS training data, all the top 10 selected features have significant differences (P-value < 0.01) between the AD and HC. Figure 4.7(b) shows a similar result on the AD2021 training data, except for the 1^{th} and 9^{th} features where no significant difference between the MCI and HC was found.



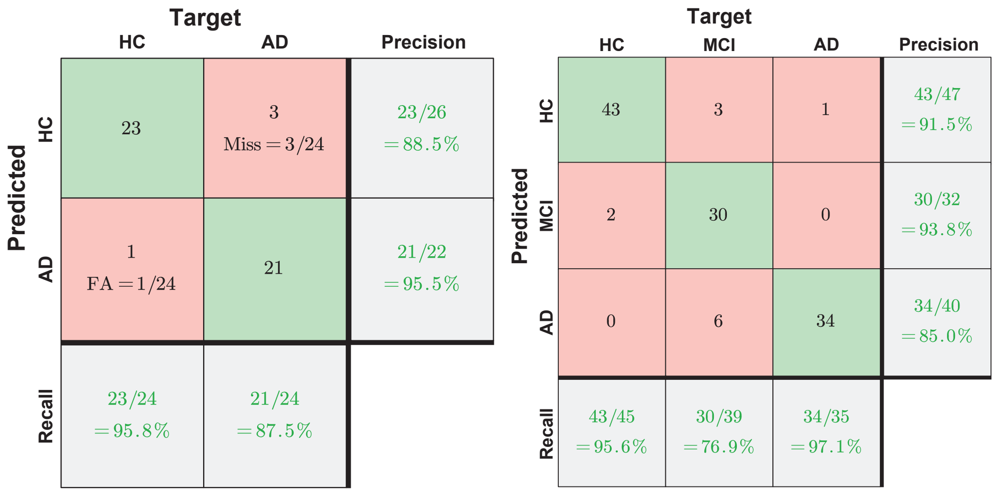
(a)

(b)

Figure 4.7: Box plots of the top 10 features selected by "MutInfo + DDR" on (a) the ADReSS and (b) the AD2021 training data. *AD*: Alzheimer's disease, *MCI*: Mild cognitive impairment, and *HC*: Healthy control. In each box, the central line represents the median, and the bottom and top edges of the box represent the 25^{th} and 75^{th} percentiles, respectively. Outliers are shown as blue '+'. The *P*-values (two-tailed Wilcoxon rank-sum test) between AD, MCI, and HC for each selected feature are given.

4.3.8 Error Analysis

To better comprehend the limitations of our proposed approach, we analyzed the subjects who were correctly or incorrectly predicted by the classifier using the features selected by our FS method. Figure 4.8 illustrates the numbers of correctly and incorrectly predicted subjects based on the test data in ADReSS and AD2021, respectively.



		Target		
		HC	AD	Precision
Predicted	HC	23	3 Miss = 3/24	23/26 = 88.5%
	AD	1 FA = 1/24	21	21/22 = 95.5%
	Recall	23/24 = 95.8%	21/24 = 87.5%	

(a)

		Target			
		HC	MCI	AD	Precision
Predicted	HC	43	3	1	43/47 = 91.5%
	MCI	2	30	0	30/32 = 93.8%
	AD	0	6	34	34/40 = 85.0%
	Recall	43/45 = 95.6%	30/39 = 76.9%	34/35 = 97.1%	

(b)

Figure 4.8: The subjects who were correctly or incorrectly predicted by the classifier using the features selected by our FS method based on the test data in (a) ADReSS and (b) AD2021. *AD*: Alzheimer's disease, *MCI*: mild cognitive impairment, *HC*: healthy control, *FA*: false alarm.

As shown in Figure 4.8(a), four subjects were incorrectly predicted (the pink boxes). In particular, a healthy subject was considered to have AD (a false alarm). Upon analyzing the transcription of this subject, we discovered that it is fairly short. Because a short transcription does not provide sufficient information for classification, it causes a false alarm. Three AD patients were considered healthy (misses). Unlike other AD patients, these patients happen to have long utterances, confusing the classifier because some linguistic features implicitly contain duration information, such as the number of words per minute (Table 4.9).

As shown in Figure 4.8(b), twelve subjects were incorrectly predicted (the pink boxes). Among the three categories (HC, MCI, and AD), subjects having MCI were the most likely to be incorrectly predicted, with nine of them being incorrectly predicted. Since MCI serves as an intermediate stage between HC and AD, the differences between HC and MCI, as well as between MCI and AD, are less evident compared to those between HC and AD. More specifically, six subjects having MCI were considered to have AD, while three subjects having MCI were considered healthy. However, these two types of incorrect predictions have different consequences in medical practices. The former misinterprets the disease progression severity, while the latter may fail to detect the onset of the disease, thereby preventing interventions to mitigate its progression at the early stage of the disease. To counteract this, we may apply a weighted loss to our FS training procedure by assigning greater weight to losses when the subjects having MCI are considered healthy. Additionally, one of the AD patients was considered healthy. A close analysis of the subject's audio revealed that while the subject was able to smoothly name several animals during the fluency test, the subject repeated some animals like "swallow" and "goat" twice. Adding repetition features to the feature set could help predicting this kind of subjects correctly.

We further analyze the performance of BERT features and pause features on the ADReSS and AD2021 datasets. Table 4.8 shows that the performance of these two feature sets on the two datasets is different. Specifically,

1) The BERT features and pause features perform well on the ADReSS dataset, thanks to the accurate manual transcriptions and precise time alignments between the transcriptions and speech recordings. Some of the pause features were selected with high selection frequency (Figure 4.6(a)).

2) In contrast, the AD2021 dataset renders the performance of these feature sets

unsatisfactory due to the erroneous automatic transcriptions. Additionally, the timestamps detected by VAD are not sufficiently accurate for extracting the pause features. Consequently, none of the pause features is among the top 100 (Figure 4.6(b)). Future work may develop a more efficient ASR system to improve the reliability of the transcriptions and investigate robust methods to mark the timestamps for speech activities.

4.4 Discussions and Conclusions

Our discussions commence with an examination of various studies on FS and its relevance to dementia detection. To identify AD patients, Haider *et al.* [15] combined various paralinguistic acoustic features – including eGeMAPS [16], ComParE 2013 [17], Emobase [17], and MRCG [18] – and applied PeaCorr tests to select the relevant features. The authors utilized PeaCorr tests to reduce the feature dimensionality of the combined feature vectors. However, the authors performed FS on the entire dataset without considering the selection frequency of individual features. In addition, they also identified the discriminative acoustic features for emotion recognition using the combined Emobase and eGeMAPS feature sets [86]. They introduced a new FS method called active feature selection (AFS) and compared its performance with other FS methods. Nevertheless, because AFS evaluates feature subsets only, it cannot measure the significance of individual features. Weiner *et al.* [55] extracted various speech-based and transcription-based features from biographic interviews to predict AD after five years. The authors utilized forward FS to reduce the size of the initial feature set. A nested leave-one-subject-out CV was performed to determine the selection frequency of individual features. However, forward FS alone cannot determine the relative importance of individual features. Additionally, nested leave-one-subject-out CV is computationally expensive for large datasets or deep-learning-based methods. Alhanai *et al.* [56] identified discriminative features from demographic, audio, and text information for cognitive impairment detection. They employed a binomial logistic regression model regularized by an elastic-net for FS. Feature importance was determined using the coefficients of the regularized logistic regression model. Nevertheless, the use of nested leave-one-subject-out CV may be impractical for large datasets.

Our study introduces enhancements to FS for dementia detection based on the above researches. For Step 1 of the two-step FS, we utilized the filter methods to pre-screen the original features. We conducted FS inside the CV instead of outside the CV, making the FS *nested* inside the learning process instead of being used as a pre-processing step. This makes individual folds select different features because the TR of individual folds are different. It is rational to nest FS inside the CV. Because if we conduct FS outside the CV, we will utilize both the TR and TS to select features and test the selected features on the TS, which will bias the performance. We adopted 10-fold CV instead of leave-one-subject-out CV for FS to avoid selection bias, as suggested by Ambroise *et al.* [87]. In the future, we will evaluate nested CV and bootstrap to see if these methods can further improve selection performance.

Our FS method has several limitations when compared with the filter methods that do not require training. For example, in the FDR, the selection variances of individual features depend on how we split the training data in the CV process. On the other hand, our FS method uses two neural networks to select features. The parameters of the trained networks depend on the initial weights and the random seed setting, causing an extra source of variation in addition to the random splits in the CV. Consequently, our method exhibits a higher selection variance.

In addition, during the CV, applying random splitting on a limited number of training samples will induce great differences across the TR. To mitigate the effect of random splittings, we propose an ensemble procedure to repeat the 10-fold CV and average the predicted scores over all the CV. For the AD2021 dataset, we divided the training samples of the same speakers into either the TR or TS to avoid selecting the features that facilitate speaker recognition instead of dementia detection. For the ADReSS dataset, because accurate manual transcriptions are provided, we prefer using transcription-based features, whereas for the AD2021 dataset, we include more speech-based features in addition to the transcription-based features because of the erroneous transcriptions.

To the best of our knowledge, this study is the first to exploit deep-learning-based methods to select spoken language biomarkers for dementia detection under limited training data scenarios. When the feature dimensionality is very large in relation to the number of training samples, the two-step FS approach can significantly reduce the feature dimensions and identify spoken language biomarkers that can achieve superior performance. Future work may investigate the biological aspects of the spoken language biomarkers.

Chapter 5

EXPERIMENTS ON DUAL-NET FEATURE RANKING (DFR)

The chapter explains the experimental setup and results on DFR. DFR is a general-purpose feature selector. To highlight its advantages, we thoroughly evaluated its capabilities on different FS tasks and different datasets. We first employed two synthetic datasets to ensure that DFR can effectively select valid features. Then, we applied DFR to the MNIST hand-written digit dataset to visualize the feature importance. These evaluations demonstrate that the proposed DFR can determine the feature relevance. Finally, we applied DFR to 12 FS benchmarks to compare its performance with some well-studied feature selectors. Finally, the DFR was evaluated on a dementia-related Cantonese corpus called JCCOCC-MoCA [88].

5.1 DFR for Feature Selection

Table 5.1 lists the characteristics and neural network parameters utilized by our proposed DFR for each dataset.

Table 5.1: The configurations and neural network parameters utilized by our proposed DFR for each dataset. The number of training iterations n is 10,000. The size of the feature mask subset $|\mathcal{Z}|$ is 32. As a result, we have 30 candidate feature mask vectors. The size of mini-batch $|\mathcal{M}|$ is 32. s is the number of selected features. p is the number of random flips.

Dataset	Number	Feature	Number	Class ratio	Dual-net feature ranking parameters			
	of samples	dimension	of classes		Operator architecture	Selector architecture	s	p
XOR	1024	10	4	0.25, 0.25, 0.25, 0.25	“10–32–32–4”	“10–32–32–1”	5	2
Binary classification	1024	10	2	0.5, 0.5	“10–32–32–2”	“10–32–32–1”	5	2
MNIST	11,982	784	2	0.51, 0.49	“784–32–32–2”	“784–1” “784–32–1” “784–32–32–1”	50	20
ALLAML	72	7,129	2	0.65, 0.35	“7,129–128–32–2”	“7,129–128–32–1”	100	20
PROSTATE_GE	102	5,966	2	0.51, 0.49	“5,966–128–32–2”	“5,966–128–32–1”	100	20
GLI_85	85	22,283	2	0.69, 0.31	“22,283–128–32–2”	“22,283–128–32–1”	100	20
LEUKEMIA	72	7,070	2	0.65, 0.35	“7,070–128–32–2”	“7,070–128–32–1”	100	20
GLIOMA	50	4,434	4	0.30, 0.28, 0.28, 0.14	“4,434–128–32–4”	“4,434–128–32–1”	100	20
CLL_SUB_111	111	11,340	3	0.46, 0.44, 0.1	“11,340–128–32–3”	“11,340–128–32–1”	100	20
COLON	62	2,000	2	0.65, 0.35	“2,000–128–32–2”	“2,000–128–32–1”	100	20
LYMPHOMA	96	4,026	9	0.48, 0.11, 0.1, 0.09, 0.06, 0.06, 0.04, 0.04, 0.02	“4,026–128–32–9”	“4,026–128–32–1”	100	20
SMK_CAN_187	187	19,993	2	0.52, 0.48	“19,993–128–32–2”	“19,993–128–32–1”	100	20
USPS	9,298	256	10	0.17, 0.14, 0.1, 0.09, 0.09, 0.09, 0.09, 0.09, 0.08, 0.08	“256–128–32–10”	“256–128–32–1”	100	20
MADELON	2,600	500	2	0.5, 0.5	“500–128–32–2”	“500–128–32–1”	100	20
ISOLET	1,560	617	26	0.038, 0.038, $\cdots$, 0.038	“617–128–32–26”	“617–128–32–1”	100	20
JCCOCC-MoCA	258	1500	2	0.5, 0.5	“1500–512–128–32–2”	“1500–512–128–32–1”	500	100

5.1.1 Implementation Details

We implemented the operator and selector using PyTorch [89], and their parameters were initialized using the PyTorch’s default initialization. Specifically, the weight matrices were initialized using the Kaiming initialization [90], while the biases were drawn from a uniform distribution. As mentioned previously, we utilized the sigmoid function as the non-linear activation function for each hidden layer of the networks. We have also tested other activation functions, e.g., ReLU, but discovered inconsistent performance.

The feature mask subset comprises $|\mathcal{Z}|-2$ candidate feature mask vectors. Therefore, enlarging $|\mathcal{Z}|$ can improve the diversity of the feature mask vectors because more candidate mask vectors are introduced. However, this improvement is accompanied by high computational complexity and GPU memory usage. In our experiments, we observed that setting $|\mathcal{Z}|$ to 32 resulted in satisfactory performance and reasonable computational cost on an RTX3090 GPU. To ensure an adequate training of the networks, we set the mini-batch size $|\mathcal{M}|$ to 32 and the number of training iterations n to 10,000.

The number of selected features s, varies across datasets. In our experiments, we set the number of flips p to 40% or 20% of s. In Section 5.1.3, we investigate the impact of varying the number of layers in the selector network on the performance. Except for the output layer, we made the number of hidden layers and number of nodes per layer identical for the operator and the selector. Additionally, for a fair comparison, we employed an identical architecture for the DFR method and other deep-learning-based methods on the same dataset.

5.1.2 Selecting Valid Features on Synthetic Data

In this subsection, we first employed two synthetic datasets in [80] to evaluate the capability of the proposed DFR in solving multidimensional XOR and non-linear binary classification problems. Because we already know the valid features and invalid features in these synthetic datasets, we evaluated whether DFR can select the valid features and ignore the invalid features. The two synthetic datasets are briefly described as follows.

- *XOR.* By grouping 8 corners of a 3-dimensional hypercube $(v_0, v_1, v_2) \in \{-1, 1\}^3$ into the tuples (v_0v_2, v_1v_2), we have 4 sets of vectors and their negations $\{\boldsymbol{v}^{(c)}, -\boldsymbol{v}^{(c)}\}_{c=1}^4$, where c is the class index. For example, the tuple $(v_0v_2, v_1v_2) = (-1, -1)$ corresponds to $c = 2$, where $\boldsymbol{v}^{(2)} = [1, 1, -1]^\mathsf{T}$. The points in class c are generated from the distribution $\frac{1}{2}\left[\mathcal{N}\left(\boldsymbol{v}^{(c)}, 0.5\boldsymbol{I}_3\right) + \mathcal{N}\left(-\boldsymbol{v}^{(c)}, 0.5\boldsymbol{I}_3\right)\right]$, where $\boldsymbol{I}_3$ is a 3×3 identity matrix and $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma})$ is a Gaussian distribution. Each sample is additionally accompanied by 7 Gaussian noise features with zero mean and unit variance, leading to a 10-dimensional feature vector.

- *Binary classification.* The points $(X_0, X_1, \ldots, X_9)$ in class $Y = -1$ are generated from the distribution $\mathcal{N}(\mathbf{0}, \boldsymbol{I}_{10})$, where $\boldsymbol{I}_{10}$ is a 10×10 identity matrix and $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma})$ is a Gaussian distribution. Given class $Y = 1$, X_0 through X_3 satisfy standard normal distribution conditioned on $9 \leq \sum_{j=0}^{3} X_j^2 \leq 16$, and $(X_4, X_5, \ldots, X_9) \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_6)$.

We thoroughly compared DFR with the following well-studied FS methods.

- *Filter methods.* The filter methods include FDR [63], PeaCorr test, and mRMR criterion [62]. These methods are usually computationally inexpensive and do not require training.

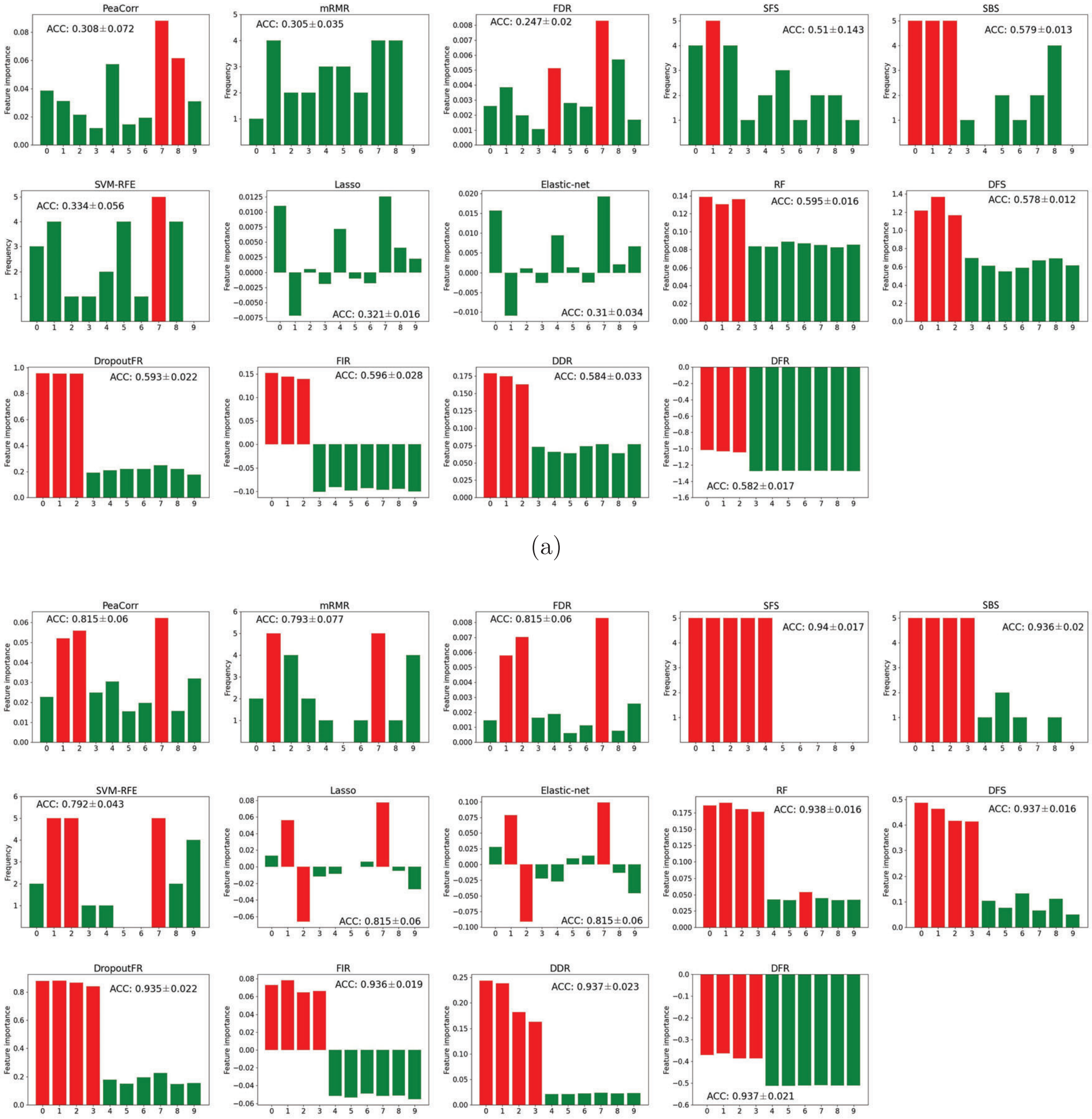
- *Wrapper methods.* The wrapper methods contain SFS and SBS, which adopt a Gaussian SVM with a box constraint of 1 to sequentially add or remove one feature in each step. We also investigated SVM-RFE [65], which ranks the coefficients of a linear SVM with a box constraint of 1 to eliminate one feature in each step.

- *Embedded methods.* The embedded methods include Lasso [59], elastic-net [57], and random forest (RF) [69]. We utilized a logistic regression model with Lasso regularization to evaluate the features' importance. The $L1$ penalty of the

Lasso regularization was set to 0.1. To implement the elastic-net, we utilized a logistic regression model with $L1$ and $L2$ regularizations. The penalty of the regularization was set to 0.1, and the mixing parameter between $L1$ and $L2$ penalties was set to 0.5. We adopted the default hyper-parameters of RF in scikit-learn to evaluate the feature importance.

- *Deep-learning-based methods.* The deep-learning-based methods include DFS [2], DropoutFR [4], FIR [3], and DDR. For these methods and DFR, we used the same network architectures and default hyper-parameters in their source codes.

We randomly generated 1024 samples for each dataset and applied 5-fold CV for evaluation. On the TR of individual folds, we applied the FS methods described above to rank and select features. As the two synthetic datasets have 3 and 4 valid features respectively, each fold selects 5 features. Based on the selected features, we adopted a Gaussian SVM with a box constraint of 1 for classification. We report the performance statistics, i.e., mean and standard deviation, of the 5-fold CV. Figure 5.1 shows the recognition accuracy and the importance of top-5 features obtained by different FS methods on the two synthetic datasets.



(a)

(b)

Figure 5.1: Recognition accuracy and feature importance obtained by different FS methods on two synthetic datasets. (a) XOR. Indexes 0–2 and 3–9 correspond to the valid and invalid features, respectively. (b) Binary classification. Indexes 0–3 and 4–9 correspond to the valid and invalid features, respectively. We adopted 5-fold CV for evaluation. Red/green colors indicate a feature selected in all of the 5 folds and in fewer than 5 folds, respectively. For example, in (a) DFR, indexes 0–2 are selected in all of the 5 folds and indexes 3–9 are selected in fewer than 5 folds. This indicates that DFR can always select the valid features because indexes 0–2 correspond to the valid features.

Figure 5.1(a) shows that on the XOR dataset, DFR, SBS, RF, DFS, DropoutFR, DDR, and FIR find the 3 valid features in all 5 folds. The filter methods, Lasso, and elastic-net, on other hand, fail to find the valid features. Figure 5.1(b) shows similar results on the binary classification dataset. The deep-learning-based methods, SBS, and RF found the 4 valid features in all 5 folds.

While SBS, RF, and deep-learning-based methods have comparable performance, their computational complexity differs. SBS is a greedy heuristic that begins with all features and iteratively removes features from the set. On the two synthetic datasets, SBS exhibits the lowest computational cost by reducing the feature dimension from 10 to 5 in just 5 iterations. However, it necessitates numerous iterations to reduce the feature dimension in high-dimensional data. Therefore, it is not applicable to problems with a large number of features. The computational complexity of RF is directly proportional to the number of trees. On the two synthetic datasets, RF has a higher computational complexity than SBS. Additionally, Figure 5.1(b) shows that RF selected an invalid feature (feature with index 6) in all 5 folds by assigning it a high feature importance. To some extent, it failed to differentiate the invalid features. The deep-learning-based methods suffer from a high computational burden due to the necessity of training deep neural networks. Among them, DFS and DropoutFR require training the parameters of a deep neural network and computing an additional weight vector. FIR, DDR, and the proposed DFR incur even higher computational costs due to the involvement of the dual-net architecture and the alternate training procedure.

5.1.3 Visualizing Feature Relevance

A good FS method should effectively determine the feature relevance. In this subsection, we evaluated whether DFR can determine the feature relevance on the MNIST hand-written digit dataset. Table 5.1 shows the characteristics of the subset.

We utilized a subset of the MNIST dataset to distinguish digits ‘3’ and ‘8’. More specifically, we flattened the 28×28 digits into 784-dimensional feature vectors. After training, we normalized and reshaped the feature importance vector $\boldsymbol{c}$ into a 28×28 matrix to represent the feature importance map. We applied 5-fold CV for evaluation. For each fold, we adopted DFR to train a dual-net and select 50 features. The selected features were then used to train a Gaussian SVM with a box constraint of 1 to classify digits '3' and '8'. To demonstrate the capability of multi-layer feature importance vector $\boldsymbol{c}$, we kept the same architecture "784–32–32–2' for the operator and visualized the feature importance maps yielded by the selector with different numbers of layers. The feature importance maps and classification results are shown in Table 5.2.

Table 5.2: Classification results and feature importance maps yielded by the selector with different numbers of layers. We kept the same architecture "784–32–32–2' for the operator. In column 3, $g(\cdot)$ indicates the sigmoid activation function. In column 4, the left pictures are the normalized feature importance maps yielded by the selector. The middle and the right pictures are the feature importance maps superimposed on the mean images of digit '3' and digit '8', respectively.

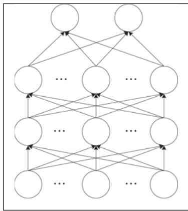
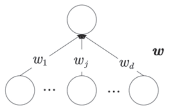
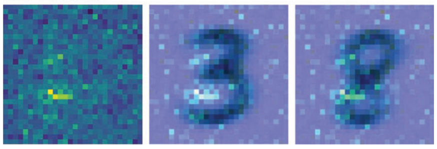
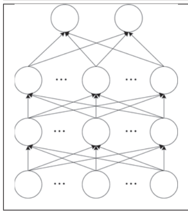
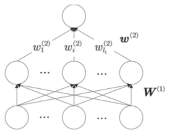
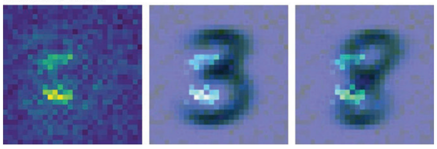
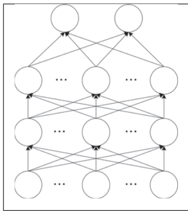
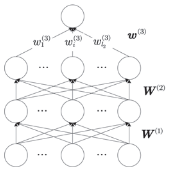
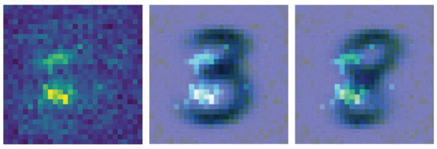
Operator	Selector	Selector architecture/ Feature importance vector $\boldsymbol{c}$	Feature importance map/ Classification accuracy (mean ± std)
		"784–1" $\boldsymbol{c} = (w_1, \ldots, w_j, \ldots, w_d)^\mathsf{T}$	
	 $w_1^{(2)}$ $w_i^{(2)}$ $w_{l_1}^{(2)}$ $\boldsymbol{w}^{(2)}$ $\boldsymbol{W}^{(1)}$	"784–32–1" $\boldsymbol{c} = g\left(\boldsymbol{W}^{(1)}\right)\boldsymbol{w}^{(2)}$	
	 $w_1^{(3)}$ $w_i^{(3)}$ $w_{l_2}^{(3)}$ $\boldsymbol{w}^{(3)}$ $\boldsymbol{W}^{(2)}$ $\boldsymbol{W}^{(1)}$	"784–32–32–1" $\boldsymbol{c} = g\left(g\left(\boldsymbol{W}^{(1)}\right)\boldsymbol{W}^{(2)}\right)\boldsymbol{w}^{(3)}$	

Table 5.2 shows that though we flatten the 28 × 28 digits into 784-dimensional feature vectors, which destroys the original spatial information, DFR can identify the relevant features. Row 1 in Table 5.2 depicts the feature importance maps yielded by the selector with one layer. The one-layer feature importance vector $\boldsymbol{c} = (w_1, \ldots, w_j, \ldots, w_d)^\mathsf{T}$ can determine the feature relevance. However, the feature importance maps in row 2 and row 3 show that the selector with more layers can better determine the relevance among the features and improve classification accuracy. This suggests that multi-layer feature importance vector $\boldsymbol{c} = g\left(\boldsymbol{W}^{(1)}\right)\boldsymbol{w}^{(2)}$ and $\boldsymbol{c} = g\left(g\left(\boldsymbol{W}^{(1)}\right)\boldsymbol{W}^{(2)}\right)\boldsymbol{w}^{(3)}$ facilitate learning the relevance among the features. Despite the complex relationship between the network's output and its input variables, the feature importance vector $\boldsymbol{c}$ provides a new way to explain the contribution of the input variables to the network's output.

5.1.4 Performance on Feature Selection Benchmarks

We further evaluated DFR on 12 FS benchmark datasets in the FS repository [1] to demonstrate its superior performance. The 12 FS benchmark datasets are described below.

- ALLAML, PROSTATE_GE, GLI_85, LEUKEMIA, GLIOMA, CLL_SUB_111, COLON, LYMPHOMA, and SMK_CAN_187 are high-dimensional biological datasets with a small number of samples, which are challenging for machine learning problems.

- USPS is a hand-written image dataset, which has 9,298 samples and 10 classes.

- MADELON is an artificial dataset for binary classification, which is part of the NIPS 2003 FS challenge.

- ISOLET is for evaluating spoken letter recognition, which has 26 classes. The dataset comprises 1,560 samples, each with 617 features.

Table 5.1 shows the details of the datasets. We applied 5-fold CV on each dataset for evaluation. According to Section 5.1.2, the strong supervised FS methods (DFS [2], DropoutFR [4], FIR [3], and DDR) can find the valid features in all folds and achieve high classification accuracy. Thus, we adopt these FS methods and DFR for comparison.[1] We used the same network architectures and default hyper-parameters in their source codes for these FS methods. In each fold, these methods select top n features, where $n = \{10, 20, \ldots, 100\}$. To evaluate the classification performance of the selected features, two classifiers were adopted, including k-NN classifier ($k = 1$) and linear SVM with a box constraint of 1.[2] As the datasets are not class-balanced, we adopted the balanced accuracy (balanced-ACC) [92] for evaluation. Figure 5.2 and Figure 5.3 show the classification performance of different numbers of selected features on the 1-NN and SVM classifiers, respectively.

[1] SBS was excluded due to its excessive computational cost on high-dimensional features. RF was excluded for its ineffective differentiation among invalid features.

[2] Classifiers' settings were adopted from [91].

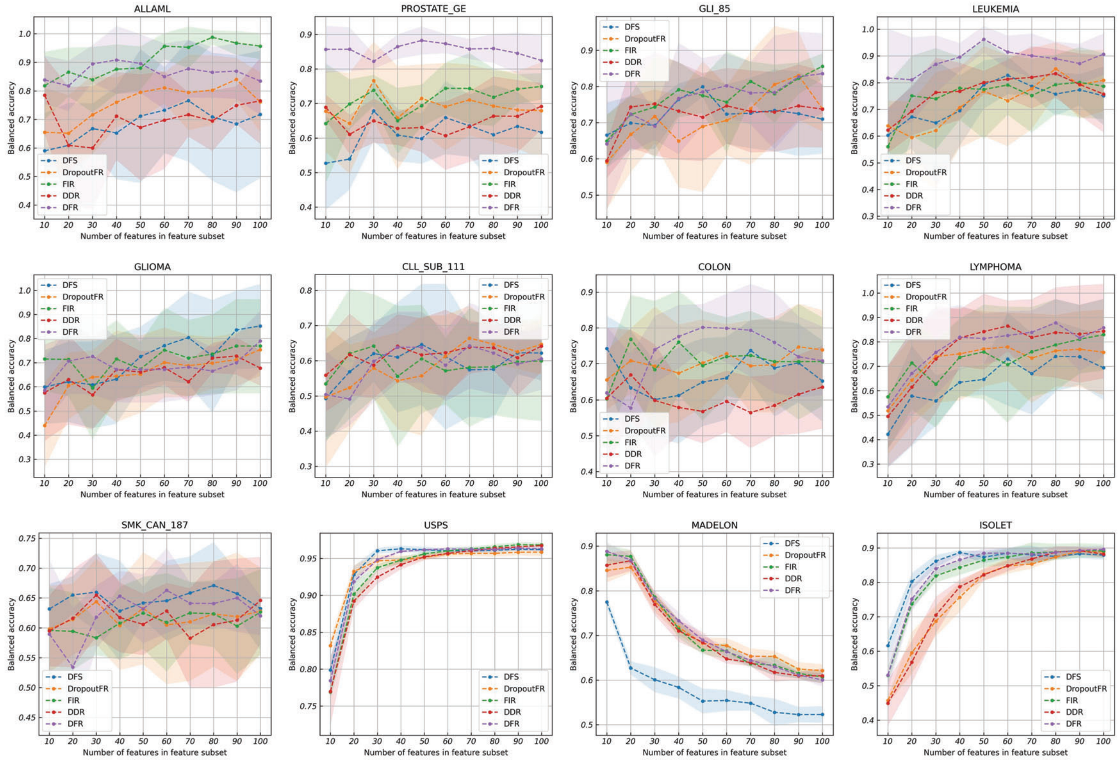


Figure 5.2: Balanced-ACC versus the number of selected features on 12 FS benchmark datasets. Features were selected by various FS methods listed in the legends, and 1-NN classifiers were used for classification. The color regions correspond to one standard deviation from the mean. *DFS*: deep feature selection [2]; *DropoutFR*: Dropout Feature Ranking [4]; *FIR*: Feature Importance Ranking [3]; *DDR*: Dual Dropout Ranking; *DFR*: Dual-net Feature Ranking.

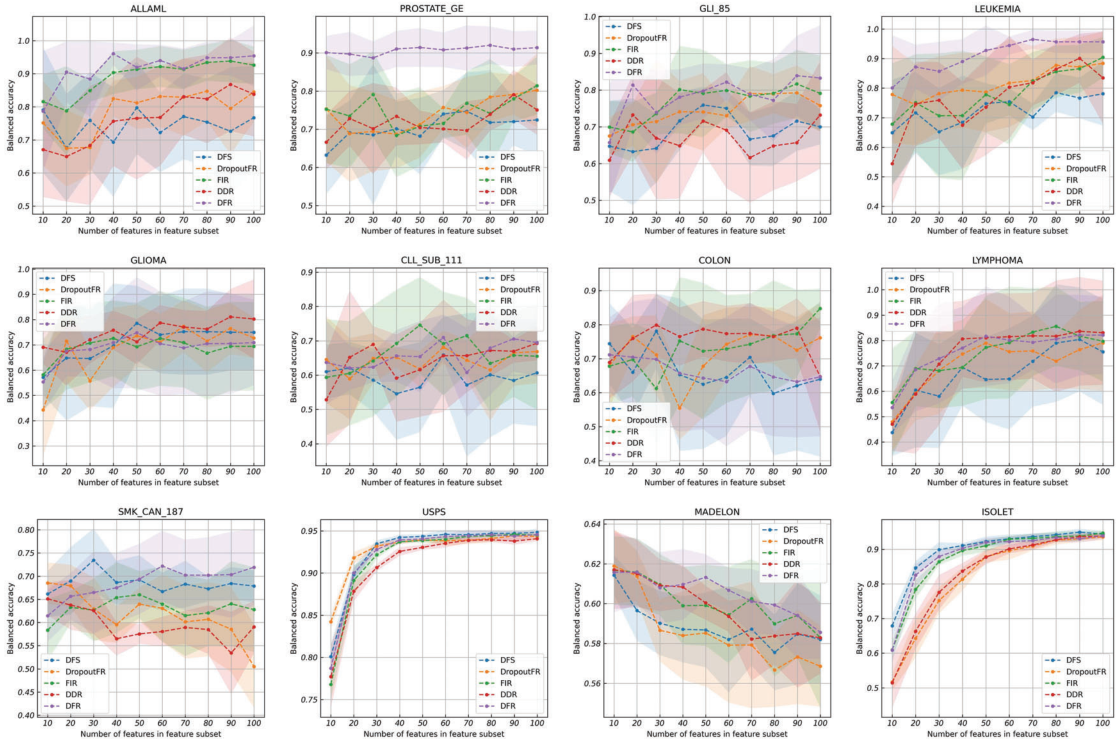


Figure 5.3: Balanced-ACC versus the number of selected features on 12 FS benchmark datasets. Features were selected by various FS methods listed in the legends, and linear SVM classifiers were used for classification. The color regions correspond to one standard deviation from the mean. *DFS*: Deep Feature Selection [2]; *DropoutFR*: Dropout Feature Ranking [4]; *FIR*: Feature Importance Ranking [3]; *DDR*: Dual Dropout Ranking; *DFR*: Dual-net Feature Ranking.

Figure 5.2 and Figure 5.3 shows that in most of the datasets, the balanced-ACC tends to increase when more features are selected. The biological datasets exhibit large fluctuation in accuracy across folds because of the small number of samples in these sets. Also, random splitting of a small dataset will lead to the samples in the individual folds having different statistics. As a result, different folds select different features and thus have fluctuated classification performance.

Table 5.3 and Table 5.4 present the average classification performance of the selected features using 1-NN and linear SVM classifiers, respectively. In the two tables, the highest average classification accuracy is highlighted in boldface and italic. Table 5.3 and Table 5.4 show that DFR performs the best on both 1-NN and linear SVM classifiers, that is, it achieves the best classification performance on most of the datasets. Additionally, on some of the biological datasets (PROSTATE_GE and LEUKEMIA), DFR performs significantly better than other methods.

Table 5.3: Average balanced-ACC of the selected features on the 1-NN classifier.

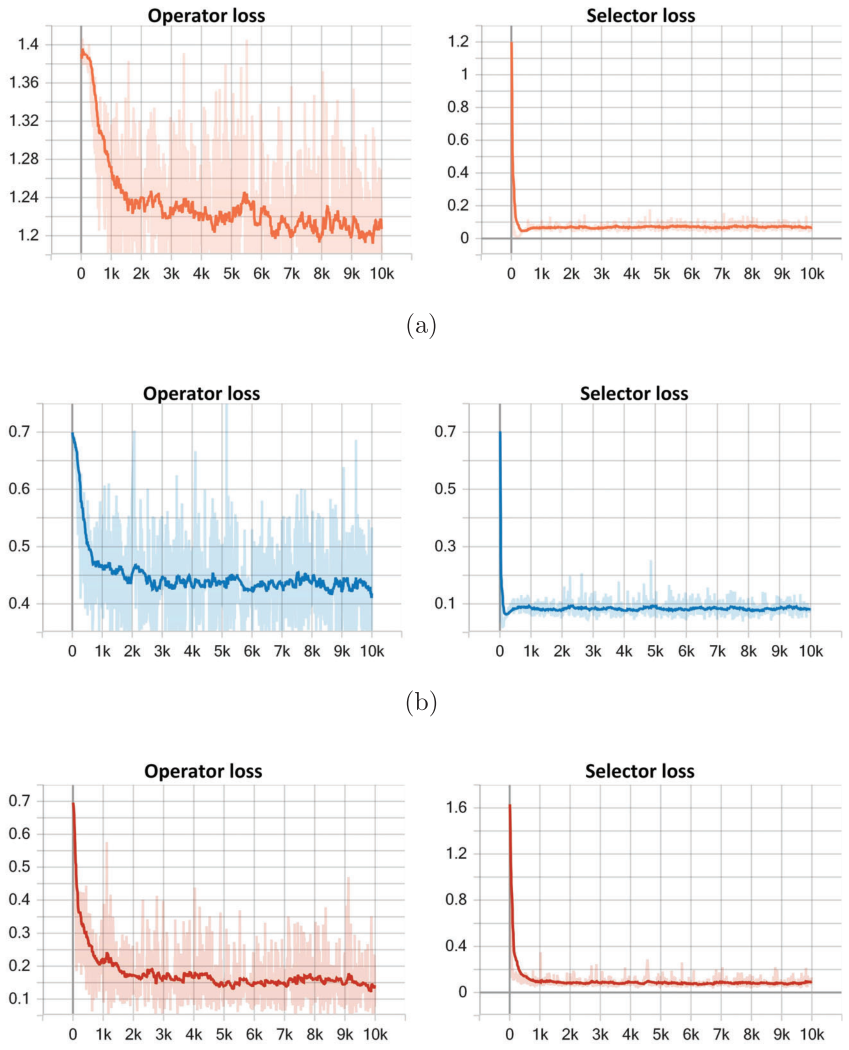
DATASET	DFS	DropoutFR	FIR	DDR	DFR
ALLAML	0.684	0.759	***0.910***	0.700	0.865
PROSTATE_GE	0.610	0.691	0.712	0.646	***0.854***
GLI_85	0.724	0.713	***0.771***	0.723	0.764
LEUKEMIA	0.731	0.729	0.753	0.767	***0.884***
GLIOMA	***0.718***	0.653	0.716	0.653	0.687
CLL_SUB_111	0.594	0.589	0.589	***0.617***	0.592
COLON	0.668	0.704	0.707	0.601	***0.729***
LYMPHOMA	0.641	0.722	0.730	0.769	***0.781***
SMK_CAN_187	***0.648***	0.618	0.609	0.616	0.624
USPS	***0.942***	0.940	0.934	0.929	0.939
MADELON	0.581	***0.711***	0.708	0.701	***0.711***
ISOLET	***0.844***	0.767	0.822	0.771	0.831
WIN	4	1	2	1	***5***

Table 5.4: Average balanced-ACC of the selected features on the linear SVM classifier.

DATASET	DFS	DropoutFR	FIR	DDR	DFR
ALLAML	0.746	0.789	0.890	0.765	***0.916***
PROSTATE_GE	0.704	0.742	0.749	0.721	***0.908***
GLI_85	0.691	0.744	0.770	0.672	***0.785***
LEUKEMIA	0.724	0.816	0.781	0.768	***0.913***
GLIOMA	0.709	0.683	0.688	***0.749***	0.688
CLL_SUB_111	0.595	0.643	***0.663***	0.642	0.659
COLON	0.667	0.716	0.732	***0.755***	0.665
LYMPHOMA	0.668	0.708	0.749	0.750	***0.758***
SMK_CAN_187	***0.685***	0.616	0.630	0.594	***0.685***
USPS	0.925	***0.927***	0.917	0.911	0.922
MADELON	0.589	0.586	0.600	0.598	***0.605***
ISOLET	***0.896***	0.820	0.875	0.828	0.879
WIN	2	1	1	2	***7***

5.1.5 Convergence of the Alternate Learning Algorithm

Our alternative learning algorithm involves the interaction between the operator net and the selector net and requires training of both networks. However, the parameters of the operator net and selector net are not updated simultaneously in each iteration. We update one network's parameters while keeping the other network's parameters fixed. We specifically demonstrate the learning behavior of the operator net and the selector net on three datasets in Figure 5.4. Figure 5.4 illustrates that regardless of datasets, the selector loss drops significantly during the first 1000 iterations and stabilizes thereafter. This indicates that the selector net can effectively predict the operator's learning performance using the selected features after 1,000 iterations. Additionally, it was observed that the feature importance yielded by the selector remains stable after around 1,000 iterations, indicating consistency in the selected features. Therefore, it can be concluded that the selector converges after 1,000 iterations. The operator loss decreases monotonically as the number of iterations increases. After the selector has converged, the operator keeps minimizing the classification loss using the selected features and eventually converges with additional iterations. Therefore, the operator converges following the selector. Our studies also show that the alternate learning algorithm enables both networks to converge on other datasets.



(a)

(b)

(c)

Figure 5.4: The train losses of the operator and selector on (a) XOR, (b) binary classification, and (c) MNIST hand-written digit datasets. The x-axis is to the number of iterations.

5.2 *Selecting Biomarkers on JCCOCC-MoCA dataset*

This section explains speech-based dementia detection and highlights various spoken language features related to dementia. The section also introduces a Cantonese dataset for speech-based dementia studies. The section finishes with a comprehensive comparison between the proposed DFR and other FS methods for spoken language biomarker selection.

5.2.1 Cantonese JCCOCC-MoCA Speech Dataset

The JCCOCC Montreal Cognitive Assessment (MoCA) Cantonese Speech corpus was collected by the CUHK Jockey Club Centre for Osteoporosis Care and Control [88]. A MoCA test [93] was given to each participant for assessing the MCI and dementia in older adults. According to the assessment results and MoCA scores, the participants were divided into four groups: (1) 205 healthy older adults; (2) 16 older adults having mild neurocognitive disorders (mild NCD); (3) 17 older adults suffering from MCI; and (4) 10 older adults suffering from major NCD.

For detecting dementia, we combined mild NCD, MCI, and major NCD into one category called possible dementia. We selected 43 healthy older adults with relatively high MoCA scores as the HC. The age distribution of the 43 selected HC and the 43 possible dementia are depicted in Figure 5.5.

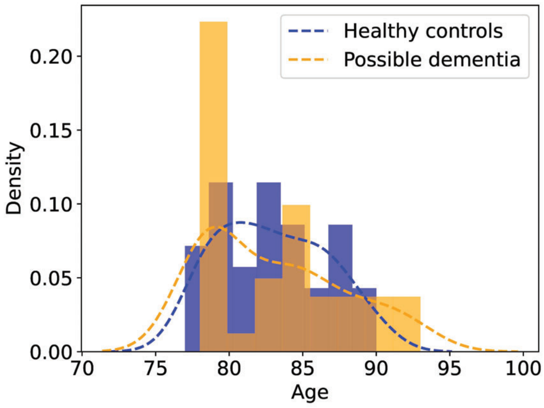


Figure 5.5: The age distribution of the 43 selected HC and the 43 possible dementia.

From the speech recording of each participant, after excluding the assessor, we extracted three 1-minute fluency tests (animals, fruits, and vegetables), resulting in 3 samples for each participant. The transcriptions corresponding to the fluency tests were also extracted. The data used for dementia detection are shown in Table 5.5.

Table 5.5: The characteristics of the JCCOCC-MoCA dataset. HC: healthy control.

Spoken languages	Cantonese
Tasks	Fluency tests (animals, fruits, and vegetables)
Number of participants	43 HC and 43 possible dementia
Number of samples	129 HC and 129 possible dementia
Manual transcriptions provided	Yes

5.2.2 Spoken Language Features

We differentiate the features into two categories: transcription-based and speech-based. The transcription-based features are described as follows.

(1) *Lexical features.* With the transcriptions of speech, the following lexical features can be extracted: the number of sentences per minute and the average number of words per sentence. Then, the PyCantonese library was utilized to parse the transcriptions.[3] After that the following features were appended to the feature set: POS counts per minute, POS ratio, the ratio of pronoun to noun, the ratio of noun to verb. These features lead to a 113-dimensional feature vector per 1-minute transcription.[4]

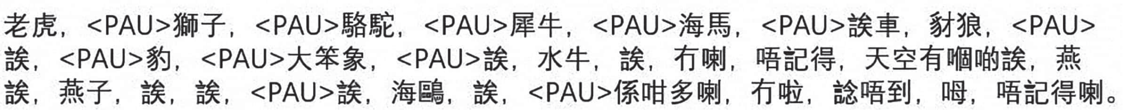
(2) *ELECTRA features.* We consider the ELECTRA model [94] pre-trained on a large Cantonese corpus as a feature extractor.[5] More specifically, we fed the transcriptions to the ELECTRA model and extracted the representations from

[3] https://pycantonese.org/

[4] Lexical features are listed in Appendix C.

the last layer of the model. For each 1-minute transcriptions, the model produces a 768-dimensional feature vector (called the ELECTRA features) that abstractly captures the semantic, syntactic, and lexical information in the transcriptions. Similar language models, e.g., BERT [26, 76] and ERNIE [27] models, have also been used for dementia detection.

(3) *Pause features.* In [77], the authors demonstrated that pauses can function as word-finding, as planning at the word, phrase, and narrative levels, and as pragmatic compensation when other interactional and narrative skills deteriorate. Thus, we included the pause features for dementia detection. In the JCCOCC-MoCA dataset, pauses and their durations have been tagged. Figure 5.6 shows an example of 1-minute transcription tagged with pauses. We divided the pauses into six groups according to their durations: G_1 (pauses between 0.05s–0.5s), G_2 (pauses between 0.5s–1s), G_3 (pauses between 1s–2s), G_4 (pauses between 2s–3s), G_5 (pauses between 3s–4s), and G_6 (pauses longer than 4s). We used the statistical characteristics of the pauses as the pause features, as illustrated in Table 5.6. For each duration group, we extracted the 5 statistical characteristics. As a result, we had a total of $5 \times 6 = 30$ pause features for each 1-minute transcription.



老虎，<PAU>獅子，<PAU>駱駝，<PAU>犀牛，<PAU>海馬，<PAU>誒車，豺狼，<PAU>
誒，<PAU>豹，<PAU>大笨象，<PAU>誒，水牛，誒，冇喇，唔記得，天空有嗰啲誒，燕
誒，燕子，誒，誒，<PAU>誒，海鷗，誒，<PAU>係咁多喇，冇啦，諗唔到，嗯，唔記得喇。

Figure 5.6: An example of 1-minute transcription tagged with pauses. The pauses are tagged as ‘<PAU>’.

Table 5.6: The 5 statistical characteristics of pauses that are extracted from 6 duration groups.

Statistical characteristic	**Description**
#p	Number of pauses per minute
%p/word ratio	Pause-to-word ratio
p duration	Total duration of pauses per minute
p mean duration	Mean duration of pauses
%p duration/word duration	Pause-duration-to-word-duration ratio

The speech-based features are further divided into five types.

(1) *Acoustic features.* We followed the standard pipelines in the COVFEFE toolbox [95] to extract the acoustic features, which include formants, loudness, pitch, zero-crossing rate, etc.

(2) *COVAREP features.* COVAREP features [41] are comprehensive acoustic features, which include prosodic features (F0 and voicing), voice quality features, and spectral features. The COVAREP features were sampled at 100Hz, and the mean, maximum, minimum, median, standard deviation, skew, and kurtosis of the features were computed, leading to a 518-dimensional feature vector per 1-minute recording. Rohanian *et al.* [96] used the COVAREP features for cognitive impairment detection.

(3) *INTERSPEECH 2010 Paralinguistic Challenge Features (IS10).* IS10 is a feature set useful for emotion recognition [97] and bipolar disorder recognition [98]. In addition to the 32 LLDs in IS09, IS10 adds 44 LLD, which include PCM loudness, 8 log Mel-frequency bands, 8 line spectral frequency pairs, F0 envelope, voicing probability, jitter, and shimmer [79]. Twelve statistics (minimum, maximum,

mean, range, etc.) of the LLD were computed, leading to a 1582-dimensional feature vector per 1-minute recording.

(4) *Emobase.* The Emobase feature set [17] comprises MFCCs, F0, F0 envelope, line spectral pairs (LSP), etc. Wang *et al.* [99] used the Emobase feature set in multi-modal attention network for AD detection.

(5) *eGeMAPS.* The extended Geneva Minimalistic Acoustic Parameter Set (eGeMAPS) [16] contains 88 features that are selected based on their potential for characterizing physiological changes in voice production.

5.2.3 Experimental Settings

We applied DFR and some other strong supervised FS methods on the JCCOCC-MoCA dataset to select spoken language biomarkers. We used different random seeds to repeat the experiments 5 times. In each experiment, we randomly grouped 68 participants into the training data and the rest 18 participants into the test data. As each participant has three 1-minute fluency tests, the training data have 204 samples, and the test data have 54 samples. The performance metrics include accuracy, precision (PRE), recall (REC), and F_1 scores with respect to the possible dementia category. We report the average performance metrics over the five repeated experiments.

5.2.4 Performance of Different Feature Types

We first evaluate the recognition performance of the full features *before* FS. On the training data, we adopted a 5-fold CV in which the samples of the same speakers were grouped into either the TR or the TS for each fold. We used a Gaussian SVM with a box constraint of 1 as the classifier to distinguish the possible dementia and the HC, as shown in Table 5.7. These results show that on the training data, the

[5] https://huggingface.co/toastynews/electra-hongkongese-base-discriminator

lexical features and ELECTRA features generally outperform the speech-based features. Additionally, the lexical features and ELECTRA features perform the best on the test data, and using all features causes a slight performance degradation. This suggests some inconsistency among the combined features.

Table 5.7: Classification performance of different feature types on the JCCOCC-MoCA dataset. The numbers in the brackets are the sizes of the feature sets. *ACC*: accuracy; *PRE*: precision; *REC*: recall.

Feature set		**5-fold CV on training data**				**Performance on test data**			
		ACC	PRE	REC	F_1	ACC	PRE	REC	F_1
Transcription-based	Lexical (113)	***0.653***	***0.662***	***0.660***	***0.642***	***0.719***	***0.726***	***0.719***	0.717
	ELECTRA (768)	0.641	0.645	0.640	0.628	***0.719***	0.721	***0.719***	***0.718***
	Pause (30)	0.519	0.537	0.535	0.513	0.604	0.607	0.604	0.601
Speech-based	Emobase (988)	0.540	0.569	0.564	0.536	0.619	0.638	0.619	0.609
	eGeMAPS (88)	0.461	0.475	0.476	0.450	0.563	0.567	0.563	0.557
	Acoustic (30)	0.508	0.518	0.516	0.490	0.589	0.594	0.589	0.586
	COVAREP (518)	0.476	0.502	0.499	0.471	0.504	0.505	0.504	0.497
	IS10 (1582)	0.511	0.533	0.529	0.504	0.581	0.585	0.581	0.578
	All features (4117)	0.621	0.631	0.625	0.609	0.715	0.724	0.715	0.712

5.2.5 Performance of Pre-screened Features

We combined all the feature sets listed in Section 5.2.2 to form 4117-dimensional feature vectors and applied a 5-fold CV on the feature vectors. On the TR of individual folds, we applied FS methods to rank and select features. The selected features were then used to train a Gaussian SVM with a box constraint of 1 to identify possible dementia.

Considering that the feature dimension is very high, filter methods were utilized to reduce the feature dimension before applying strong supervised FS methods. On the TR of individual folds, we applied filter methods, including PeaCorr, FDR, and MutInfo to reduce the feature dimension from 4117 to $\{250, 500, 1000, 1500, 2500, 3500\}$, as shown in Table 5.8. Table 5.8 shows that using filter methods to pre-screen features can reduce the inconsistency and redundancy among the combined features and improve recognition performance. By using PeaCorr to reduce the feature dimension to 1500, we obtained the best CV performance on the training data. Therefore, on the TR of individual folds, subsequent experiments utilized PeaCorr to reduce the feature dimension to 1500.

Table 5.8: Classification accuracy of different numbers of features selected by filter methods on the JCCOCC-MoCA training data.

Feature dimension	**5-fold CV on training data**		
	FDR	PeaCorr	MutInfo
250	0.639 ± 0.050	0.631 ± 0.049	0.623 ± 0.037
500	0.621 ± 0.042	0.622 ± 0.047	0.632 ± 0.041
1,000	0.637 ± 0.038	0.639 ± 0.036	0.635 ± 0.041
1,500	0.648 ± 0.048	$\mathbf{0.648 \pm 0.046}$	0.640 ± 0.032
2,500	0.639 ± 0.052	0.638 ± 0.053	0.628 ± 0.039
3,500	0.626 ± 0.051	0.627 ± 0.045	0.627 ± 0.041

5.2.6 Performance of Deep-learning-based Methods

We report the performance of DFR and some other strong supervised FS methods on the JCCOCC-MoCA dataset. These FS methods include DFS [2], DropoutFR [4], FIR [3], and DDR. We used the same network architectures for all the methods. Additionally, on the training data, we determined the best possible hyper-parameter settings for each method using grid-search and CV. For a fair comparison, during the leave-n-subject out CV, we selected 500 features in each fold. The recognition accuracy of the feature subsets identified by different methods are shown in Table 5.9.

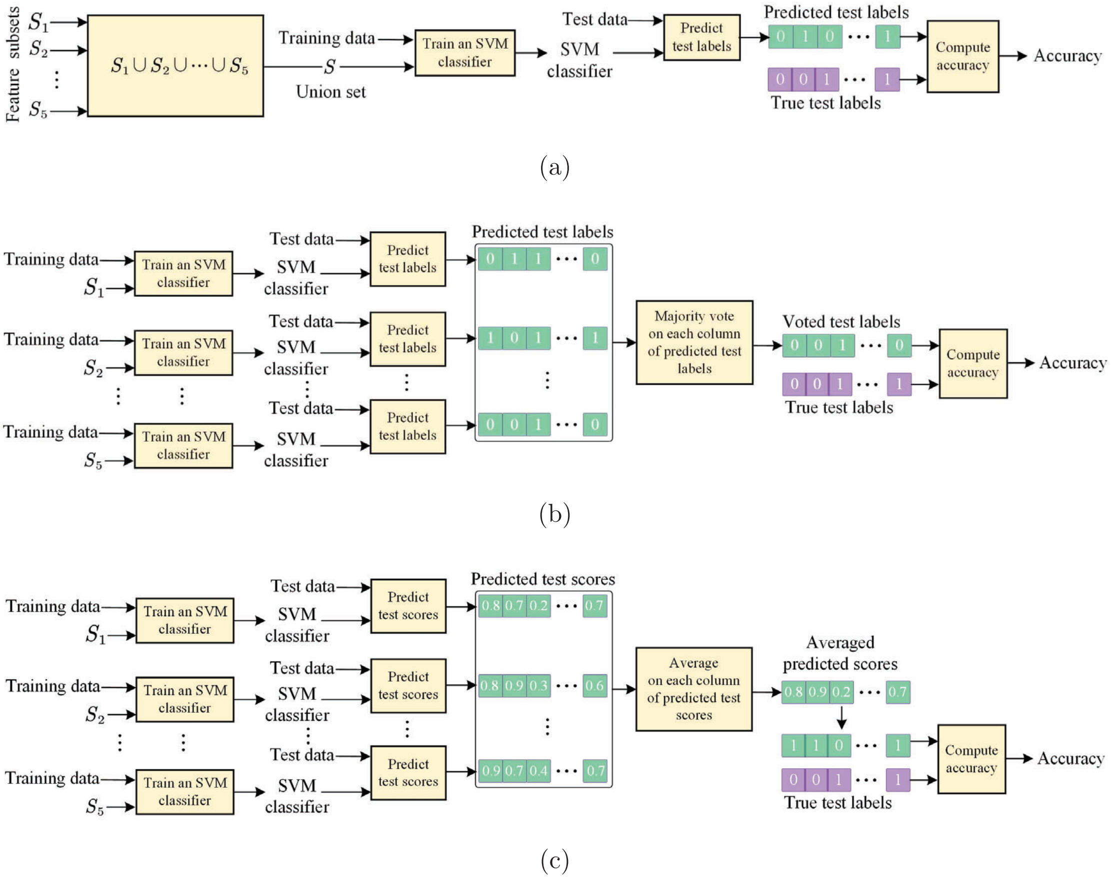
Table 5.9: Recognition accuracy of the feature subsets identified by different deep-learning-based methods on the JCCOCC-MoCA training data. The size of the feature subsets is 500. The results are presented as the mean ± standard deviation of the five repeated experiments. ACC: accuracy; PRE: precision; REC: recall.

Method	**5-fold CV on training data**			
	ACC	PRE	REC	F_1
DFS [2]	**0.663** ± 0.038	**0.664** ± 0.034	**0.663** ± 0.031	**0.652** ± 0.040
DropoutFR [4]	0.650 ± 0.041	0.654 ± 0.036	0.651 ± 0.033	0.638 ± 0.042
FIR [3]	**0.663** ± 0.033	**0.664** ± 0.030	0.661 ± 0.030	0.650 ± 0.036
DDR	0.638 ± 0.038	0.640 ± 0.029	0.637 ± 0.029	0.627 ± 0.038
DFR	0.651 ± 0.042	0.657 ± 0.037	0.652 ± 0.033	0.639 ± 0.043

Table 5.9 shows that all the methods achieve superior performance compared with the combined features. Additionally, though it reduces the feature dimension from 1500 to 500, DFS, DropoutFR, FIR, and DFR further improve classification performance compared with PeaCorr. DFS and FIR achieve the best performance on the training data.

5.2.7 Performance on Test Data

We then evaluated the identified feature subsets on the test data. Because each fold uses different TR for training, the feature relevance in different folds is not the same, and different folds will produce different feature subsets. We propose three techniques to fuse the different feature subsets so that high performance can be achieved on the test data. They are the union of feature subsets (Tech 1), majority voting on predicted labels (Tech 2), and soft voting on predicted scores (Tech 3), as shown in Figure 5.7.



(a)

(b)

(c)

Figure 5.7: Two techniques are used to fuse the feature subsets identified by the leave-n-subject out CV. (a) Union of feature subsets (Tech 1). (b) Majority voting on predicted labels (Tech 2). (c) Soft voting on predicted scores (Tech 3).

Tech 1 first takes the union of feature subsets identified by the leave-n-subject out CV; and then performs prediction on the union set. Tech 2 obtains the predicted labels using each of the feature subsets; then it applies majority voting on the predicted labels to make the final decisions. Tech 3 obtains the predicted scores using each of the feature subsets; then it averages the predicted scores to make the final decisions. Additionally, we compared our selected features with speech-based embeddings extracted from pre-trained ASR models, including Wav2vec 2.0[6], Whisper[7], HuBERT[8], and WavLM[9]. Given N speech frames of an utterance, we extract the embeddings from the last hidden layer's output of the embedding network. This operation results in a $D \times N$ hidden state matrix, where D is the number of hidden nodes in the last hidden layer. The speech-based embeddings are then obtained by averaging the hidden state matrix across the N frames. Our approach is similar to the AD recognition task in [20], where the hidden state matrix goes through a convolution layer before being averaged across the N frames.

The recognition accuracy of speech-based embeddings and the feature subsets on the test data are shown in Table 5.10. Table 5.10 shows that our selected feature subsets generally outperform the speech-based embeddings on the JCCOCC-MoCA test data. Table 5.10 also shows that based on the identified feature subsets, all the methods achieve comparable performance compared with the combined features. Additionally, after utilizing Tech 1, Tech 2, and Tech3 to fuse the feature subsets, we obtain better performance on the test data. After utilizing Tech 2, DFR achieves the best performance on the test data and outperforms the combined features by around 3% in terms of accuracy. This reasonable gain only relies on some small feature subsets. Although DFS and FIR achieve the best performance on the training data, they do not perform the best on the test data. This contradictory result reflects that DFS and FIR are prone to over-fitting on the JCCOCC-MOCA training data.

[6] https://huggingface.co/facebook/wav2vec2-large-xlsr-53

[7] https://huggingface.co/openai/whisper-large

[8] https://huggingface.co/facebook/hubert-large-ll60k

[9] https://huggingface.co/microsoft/wavlm-large

Table 5.10: Recognition accuracy of speech-based embeddings and the feature subsets identified by different deep-learning-based methods on the JCCOCC-MoCA test data. During the 5-fold CV, five feature subsets were selected, one for each fold. When evaluating the feature subsets on the test data, we obtained one accuracy for each feature subset and then averaged the accuracy. *TECH 1*: Union of feature subsets; *TECH 2*: Majority voting on predicted labels; *TECH 3*: Soft voting on predicted scores. See Figure 5.7 for details. The results are presented as the mean ± standard deviation of the five repeated experiments. *ACC*: accuracy; *PRE*: precision; *REC*: recall.

		Performance on test data			
		ACC	PRE	REC	F_1
Speech-based embeddings	Wav2vec 2.0 [23]	0.659 ± 0.063	0.664 ± 0.064	0.659 ± 0.063	0.656 ± 0.063
	HuBERT [100]	0.578 ± 0.077	0.581 ± 0.081	0.578 ± 0.077	0.574 ± 0.077
	WavLM [101]	0.585 ± 0.052	0.592 ± 0.056	0.585 ± 0.052	0.580 ± 0.050
	Whisper [102]	0.633 ± 0.067	0.637 ± 0.069	0.633 ± 0.067	0.630 ± 0.069
FS	DFS [2]	0.714 ± 0.047	0.719 ± 0.045	0.714 ± 0.047	0.712 ± 0.049
	DropoutFR [4]	0.710 ± 0.038	0.716 ± 0.036	0.710 ± 0.038	0.708 ± 0.040
	FIR [3]	0.707 ± 0.049	0.712 ± 0.048	0.707 ± 0.049	0.705 ± 0.051
	DDR	0.719 ± 0.047	0.724 ± 0.045	0.719 ± 0.047	0.717 ± 0.048
	DFR	0.711 ± 0.042	0.716 ± 0.041	0.711 ± 0.042	0.709 ± 0.043
FS + Tech 1	DFS [2]	0.733 ± 0.051	0.737 ± 0.048	0.733 ± 0.051	0.732 ± 0.052
	DropoutFR [4]	0.733 ± 0.040	0.738 ± 0.039	0.733 ± 0.040	0.732 ± 0.041
	FIR [3]	0.726 ± 0.038	0.729 ± 0.036	0.726 ± 0.038	0.725 ± 0.039
	DDR	0.730 ± 0.053	0.734 ± 0.051	0.730 ± 0.053	0.728 ± 0.054
	DFR	0.726 ± 0.030	0.732 ± 0.029	0.726 ± 0.030	0.724 ± 0.030
FS + Tech 2	DFS [2]	0.722 ± 0.048	0.727 ± 0.046	0.722 ± 0.048	0.720 ± 0.050
	DropoutFR [4]	0.730 ± 0.048	0.733 ± 0.044	0.730 ± 0.048	0.728 ± 0.049
	FIR [3]	0.719 ± 0.066	0.722 ± 0.063	0.719 ± 0.066	0.717 ± 0.068
	DDR	0.737 ± 0.050	0.740 ± 0.049	0.737 ± 0.050	0.736 ± 0.051
	DFR	$\mathbf{0.744} \pm 0.066$	$\mathbf{0.747} \pm 0.064$	$\mathbf{0.744} \pm 0.066$	$\mathbf{0.743} \pm 0.066$
FS + Tech 3	DFS [2]	0.733 ± 0.036	0.738 ± 0.036	0.733 ± 0.036	0.732 ± 0.037
	DropoutFR [4]	0.733 ± 0.038	0.737 ± 0.037	0.733 ± 0.038	0.732 ± 0.039
	FIR [3]	0.730 ± 0.051	0.734 ± 0.047	0.730 ± 0.051	0.728 ± 0.052
	DDR	0.722 ± 0.031	0.725 ± 0.030	0.722 ± 0.031	0.721 ± 0.032
	DFR	0.726 ± 0.045	0.732 ± 0.044	0.726 ± 0.045	0.724 ± 0.045

5.2.8 Analyzing the Selected Features

We finally depict 150 features selected by DFR with the highest selection frequency in Figure 5.8. Figure 5.8 shows that the ELECTRA features prevail in the top 150. This is reasonable because the ELECTRA features perform well on the training data. However, though the lexical features perform the best on the training data, only a few of them were selected. This suggests that not all features in a feature group are selected even though the feature group is good for classification. Interestingly, though the speech-based feature groups have unsatisfactory performance, some features in the speech-based feature groups were selected. This suggests that in addition to the high-level semantic and syntactic information obtainable from the transcriptions, there is also low-level acoustic information that is predictive of dementia. This finding is in line with the literature [53, 55, 56], where in addition to the transcription-based features, low-level speech-based anomaly are also indicative of dementia.

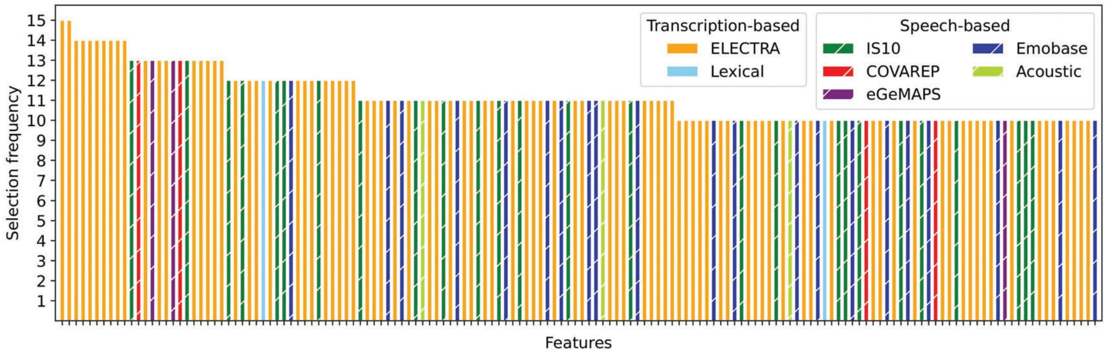


Figure 5.8: 150 features selected by DFR with the highest selection frequency. The maximum selection frequency is 25 because we repeated the experiment 5 times, with a 5-fold CV for each experiment.

Through a meticulous examination of the selected features, we found that some of them are closely correlated with previous research, as shown in Table 5.11. Although Alhanai *et al.* [56] focused on discriminating English-speaking patients with MCI while we focus on discriminating Cantonese-speaking patients with NCD, we share some of the selected features. The feature sharing indicates that despite the differences in the patient populations and disease severity, some speech anomalies are presented in both groups of patients.

Table 5.11: Some of the features selected by our proposed method are closely correlated with previous research in the same field.

Selected feature	Known specificity
The differential frame-to-frame jitter	Alhanai *et al.* [56] revealed that decreasing *jitter* is positively correlated with MCI.
The mean of voice segment lengths per second	Alhanai *et al.* [56] revealed that shorter *speech segment lengths* is positively correlated with MCI.
Total word types	Our previous research also showed that the "total word types" was frequently selected in distinguishing AD patients in the ADReSS English corpus [103].
The ratio of nouns	Jarrold *et al.* [83] observed a decrease in *the proportion of nouns* among AD patients.
The ratio of nouns to verbs	AD patients may experience more difficulty in naming verbs as compared to nouns [39].

5.3 Discussions and Conclusions

We conducted FS inside the CV instead of outside the CV, making the FS *nested* inside the learning process instead of being used as a pre-processing step. This makes individual folds select different features because the TR of individual folds are different. It is rational to nest FS inside the CV. This is because if we conduct FS outside the CV, we will utilize both the TR and TS to select features and test the selected features on the TS, which will bias the performance.

In conclusion, we utilized a dual-net architecture along with an alternate learning algorithm for FS. The method uses the network's parameters to explain the contribution of the input variables to the prediction of the deep neural network. This explains the feature importance of individual variables and also allows for learning the relevance among the variables. Thorough evaluations on the synthetic, MNIST hand-written digit, and FS benchmark datasets manifest that the proposed method outperforms several state-of-the-art supervised FS methods.

On the JCCOCC-MoCA dataset, we divided the training samples of the same speakers into either the TR or TS to avoid selecting the features that facilitate speaker recognition instead of dementia detection. The spoken language biomarkers selected by the method achieve comparable or supervisor performance compared with the combined features. Future work may investigate the biological aspects of the spoken language biomarkers.

Chapter 6

CONCLUSIONS AND FUTURE PERSPECTIVE

Investigating the detection of dementia through spontaneous speech is an important research area with potential contributions to human health and well-being. Effective detection of early signs of the disease can facilitate timely intervention to slow deterioration. Our study not only compares different feature sets but also aims to automatically select the discriminative features, referred to as spoken language biomarkers, for dementia detection. We evaluated our proposed FS methods on three dementia-related corpora from different spoken languages, including ADReSS (English), AD2021 (Mandarin Chinese), and JCCOCC-MoCA (Cantonese). Due to the language-specific nature of transcription-based features, we employed different extraction methods for different languages. For example, to extract the linguistic features from the ADReSS dataset, we employed the EVAL command in the CLAN program to parse the English transcriptions; on the other hand, when extracting the lexical features from the JCCOCC-MoCA dataset, we utilized the PyCantonese library to parse the Cantonese transcriptions. Extracting transcription-based features for Cantonese is more challenging due to the limited research on linguistic features in Cantonese compared to English. Consequently, we could only extract lexical features from the JCCOCC-MoCA Cantonese corpus instead of linguistic features. Future work may investigate the linguistic features specific to Cantonese to improve detection performance.

The linguistic features for dementia detection are language-specific and can be categorized into lexical, syntactic, and semantic features. This study utilized three datasets on different languages. For the ADReSS English dataset, given the extensive research on dementia detection using English corpora, we could easily employ the EVAL command in the CLAN program to extract the well-defined linguistic features. Conversely, for the JCCOCC-MoCA Cantonese dataset, due to the limited research on dementia detection using Cantonese corpora, expertise in extracting linguistic features was scarce. For dementia detection on the Cantonese dataset, we initially referred to research conducted on the English dataset. Our approach involved extracting linguistic features by adapting the English's feature extraction pipeline, resulting in the extraction of lexical features. However, further investigations are necessary to extract the language-specific linguistic features for Cantonese.

When extracting lexical features from the AD2021 dataset, we utilized an automated POS tagger, the Stanford POS tagger, to parse the Chinese transcriptions. The performance of lexical features is influenced by the POS tagger because the extraction of lexical features relies on POS tagging. POS tagging in Mandarin Chinese presents greater challenges compared to English, as the same words may have varying POS tags depending on the context. We observed that the lexical features in the AD2021 dataset perform much worse than linguistic features in the ADReSS dataset. This finding could be attributed to the erroneous automatic transcriptions in the AD2021 dataset and the unsatisfied performance of the Stanford POS tagger. To improve the performance of the lexical features, future work may improve ASR performance and implement advanced POS taggers. For example, deep-learning-based POS taggers, trained on extensive corpora, can better utilize contextual information for POS tagging. Similarly, in the JCCOCC-MoCA Cantonese dataset, replacing the PyCantonese POS tagger with advanced alternatives may improve the performance of the lexical features.

This study computed the statistics of different durations of *silent pauses* as the pause features. This was inspired from the potential of disfluencies to signify dementia. Yuan *et al.* [32] encoded different durations of silent pauses using three punctuations `<.>`, `<..>`, and `<...>` to emphasize the disfluencies in the transcriptions. Results show that adding the silent pauses can substantially improve the performance of AD detection. Our study considered the silent pause features as candidates for FS to identify which durations of silent pauses are indicative of dementia, and results are shown in Table 4.9. Additionally, *filled pauses* could also be disfluencies of the spoken language. Our study has also investigated the statistics of filled pauses, such as the proportion of filled pauses, as features to detect dementia. However, experimental results did not demonstrate their efficiency. One possible explanation is that filled pauses may not be reliable indicators of dementia, as healthy individuals could also use them in their speech. Another possible explanation could be that simply using the statistics of filled pauses does not indicate dementia; instead, atypical patterns of filled pauses, such as their misplacement within sentences by dementia patients, may provide better insights. Future work may explore these atypical patterns and evaluate their efficiency for dementia detection.

Detecting dementia from spontaneous speech presents a challenge due to the constraints of small sample size resulting from time-consuming data collection. Given the limited number of samples, we need to address the data scarcity issue. For instance, the proposed FS method requires training dual neural networks on a limited number of samples, which leads to variations in the FS results across different runs. In such cases, we ensured the reproducibility of FS results by using the same random seed for each run. More importantly, we performed FS within CV using multiple data splittings to obtain the selection frequencies. The selection frequencies are meaningful as it offers insights into the consistency and robustness of the selected features, enabling us to assess their reliability across various iterations and data splittings. During the evaluation on the test data, apart from assessing the classification performance of each feature subset individually, we fused the feature subsets prior to obtaining their performance on the test data. The fusion not only addresses the issue of evaluating the performance of multiple feature subsets on the test data but also improves the detection performance on the test data.

Our study focuses on selecting spoken language biomarkers to detect dementia. However, the correlation between the biomarkers and the nature of dementia is weak. Future work needs to establish the biological relevance of the selected biomarkers to brain functions and the nature of the disease. Additionally, the entire process is not entirely automated. For example, extracting linguistic and lexical features necessitates manual transcriptions. Extracting BERT or ELECTRA features also necessitates manual transcriptions. Otherwise, their classification performance may be compromised by erroneous automatic transcriptions. Implementing our methods to screen the disease in large populations is impractical due to the labor-intensive nature of annotations and manual transcriptions. In future work, it would be meaningful to research the automatic recognition of elderly speech so that accurate and fully automatic dementia screening systems can become practical.

BIBLIOGRAPHY

[1] J. Li, K. Cheng, S. Wang, F. Morstatter, R. P. Trevino, J. Tang, and H. Liu, "Feature selection: A data perspective," *ACM Comput. Surv.*, vol. 50, no. 6, pp. 1–45, Dec. 2017.

[2] Y. Li, C. Y. Chen, and W. W. Wasserman, "Deep feature selection: Theory and application to identify enhancers and promoters," *J. Comput. Biol.*, vol. 23, no. 5, pp. 322–336, May 2016.

[3] M. Wojtas and K. Chen, "Feature importance ranking for deep learning," in *Proc. Adv. neural inf. proces. syst. (NIPS)*, Oct. 2020, pp. 5105–5114.

[4] C. H. Chang, L. Rampasek, and A. Goldenberg, "Dropout feature ranking for deep learning models," 2017. [Online]. Available: https://arxiv.org/abs/1712.08645

[5] J. L. Cummings, R. Doody, and C. Clark, "Disease-modifying therapies for Alzheimer disease: Challenges to early intervention," *Neurology*, vol. 69, no. 16, pp. 1622–1634, Oct. 2007.

[6] K. A. Johnson, N. C. Fox, R. A. Sperling, and W. E. Klunk, "Brain imaging in Alzheimer disease," *Cold Spring Harb. Perspect. Med.*, vol. 2, no. 4, p. a006213, Jan. 2012.

[7] C.-C. Liu, T. Kanekiyo, H. Xu, and G. Bu, "Apolipoprotein E and Alzheimer disease: risk, mechanisms and therapy," *Nat. Rev. Neurol.*, vol. 9, no. 2, pp. 106–118, Jan. 2013.

[8] J.-H. Song, J.-T. Yu, and L. Tan, “Brain-derived neurotrophic factor in Alzheimer’s disease: Risk, mechanisms, and therapy,” *Mol. Neurobiol.*, vol. 52, no. 3, pp. 1477–1493, Oct. 2014.

[9] L. M. Shaw, J. Arias, K. Blennow, D. Galasko, J. L. Molinuevo, S. Salloway, S. Schindler, M. C. Carrillo, J. A. Hendrix, A. Ross, J. Illes, C. Ramus, and S. Fifer, “Appropriate use criteria for lumbar puncture and cerebrospinal fluid testing in the diagnosis of Alzheimer’s disease,” *Alzheimers. Dement.*, vol. 14, no. 11, pp. 1505–1521, Oct. 2018.

[10] L. Mickes, J. T. Wixted, C. Fennema-Notestine, D. Galasko, M. W. Bondi, L. J. Thal, and D. P. Salmon, “Progressive impairment on neuropsychological tasks in a longitudinal study of preclinical Alzheimer’s disease.” *Neuropsychology*, vol. 21, no. 6, pp. 696–705, Nov. 2007.

[11] D. Beltrami, L. Calzà, G. Gagliardi, E. Ghidoni, N. Marcello, R. R. Favretti, and F. Tamburini, “Automatic identification of mild cognitive impairment through the analysis of Italian spontaneous speech productions,” in *Proc. Int. Conf. Lang. Resour. Eval. (LREC)*, May 2016, pp. 2086–2093.

[12] J. J. G. Meilán, F. Martínez-Sánchez, J. Carro, D. E. López, L. Millian-Morell, and J. M. Arana, “Speech in Alzheimer’s disease: Can temporal and acoustic parameters discriminate dementia?” *Dement. Geriatr. Cogn. Disord.*, vol. 37, no. 5–6, pp. 327–334, Jun. 2014.

[13] K. L. de Ipiña, J. B. Alonso, J. Solé-Casals, N. Barroso, P. Henriquez, M. Faundez-Zanuy, C. M. Travieso, M. Ecay-Torres, P. Martínez-Lage, and H. Eguiraun, “On automatic diagnosis of Alzheimer’s disease based on spontaneous speech analysis and emotional temperature,” *Cogn. Comput.*, vol. 7, no. 1, pp. 44–55, Aug. 2013.

[14] A. König, A. Satt, A. Sorin, R. Hoory, O. Toledo-Ronen, A. Derreumaux, V. Manera, F. Verhey, P. Aalten, P. H. Robert, and R. David, "Automatic speech analysis for the assessment of patients with predementia and Alzheimer's disease," *Alzheimers Dement.-Diagn. Assess. Dis. Monit.*, vol. 1, no. 1, pp. 112–124, Mar. 2015.

[15] F. Haider, S. de la Fuente, and S. Luz, "An assessment of paralinguistic acoustic features for detection of Alzheimer's dementia in spontaneous speech," *IEEE J. Sel. Top. Signal Process.*, vol. 14, no. 2, pp. 272–281, Feb. 2020.

[16] F. Eyben, K. R. Scherer, B. W. Schuller, J. Sundberg, E. Andre, C. Busso, L. Y. Devillers, J. Epps, P. Laukka, S. S. Narayanan, and K. P. Truong, "The Geneva minimalistic acoustic parameter set (GeMAPS) for voice research and affective computing," *IEEE Trans. Affect. Comput.*, vol. 7, no. 2, pp. 190–202, Apr. 2016.

[17] F. Eyben, M. Wöllmer, and B. Schuller, "openSMILE - the Munich versatile and fast open-source audio feature extractor," in *Proc. ACM Multimedia Int. Conf.*, Oct. 2010, pp. 1459–1462.

[18] F. Haider and S. Luz, "Attitude recognition using multi-resolution cochleagram features," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, May 2019, pp. 3737–3741.

[19] S. Nasreen, J. Hough, and M. Purver, "Detecting Alzheimer's disease using interactional and acoustic features from spontaneous speech," in *Proc. Interspeech*, Aug. 2021, pp. 1962–1966.

[20] L. Gauder, L. Pepino, L. Ferrer, and P. Riera, "Alzheimer disease recognition

using speech-based embeddings from pre-trained models," in *Proc. Interspeech*, Aug. 2021, pp. 3795–3799.

[21] J. Shor, A. Jansen, R. Maor, O. Lang, O. Tuval, F. d. C. Quitry, M. Tagliasacchi, I. Shavitt, D. Emanuel, and Y. Haviv, "Towards learning a universal non-semantic representation of speech," 2020. [Online]. Available: https://arxiv.org/abs/2002.12764

[22] X. Li, S. Dalmia, J. Li, M. Lee, P. Littell, J. Yao, A. Anastasopoulos, D. R. Mortensen, G. Neubig, A. W. Black, and F. Metze, "Universal phone recognition with a multilingual allophone system," in *Proc. IEEE Int. Conf. Acoust, Speech Signal Process. (ICASSP)*, May 2020, pp. 8249–8253.

[23] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "Wav2vec 2.0: A framework for self-supervised learning of speech representations," in *Proc. Advances Neural Inf. Process. Syst. (NIPS)*, vol. 33, 2020, pp. 12 449–12 460.

[24] A. Balagopalan and J. Novikova, "Comparing acoustic-based approaches for Alzheimer's disease detection," in *Proc. Interspeech*, Aug. 2021, pp. 3800–3804.

[25] Y. Qiao, X. Yin, D. Wiechmann, and E. Kerz, "Alzheimer's disease detection from spontaneous speech through combining linguistic complexity and (dis)fluency features with pretrained language models," in *Proc. Interspeech*, Aug. 2021, pp. 3805–3809.

[26] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," 2018. [Online]. Available: https://arxiv.org/abs/1810.04805

[27] Y. Sun, S. Wang, Y. Li, S. Feng, H. Tian, H. Wu, and H. Wang, "ERNIE 2.0:

A continual pre-training framework for language understanding," *Proc. AAAI Conf. Artif. Intell.*, vol. 34, no. 05, pp. 8968–8975, Apr. 2020.

[28] Z. S. Syed, M. S. S. Syed, M. Lech, and E. Pirogova, "Tackling the ADRESSO challenge 2021: The MUET-RMIT system for Alzheimer's dementia recognition from spontaneous speech," in *Proc. Interspeech*, Aug. 2021, pp. 3815–3819.

[29] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter," 2019. [Online]. Available: https://arxiv.org/abs/1910.01108

[30] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "RoBERTa: A robustly optimized BERT pretraining approach," 2019. [Online]. Available: https://arxiv.org/abs/1907.11692

[31] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, "BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," in *Proc. Annu. Meet. Assoc. Comput Linguist.*, 2020, pp. 7871–7880.

[32] J. Yuan, Y. Bian, X. Cai, J. Huang, Z. Ye, and K. Church, "Disfluencies and fine-tuning pre-trained language models for detection of Alzheimer's disease." in *Proc. Interspeech*, Oct. 2020, pp. 2162–2166.

[33] C. Li, D. Knopman, W. Xu, T. Cohen, and S. Pakhomov, "GPT-D: Inducing dementia-related linguistic anomalies by deliberate degradation of artificial neural language models," in *Proc. Annu. Meet. Assoc. Comput Linguist.*, May 2022, pp. 1866–1877.

[34] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever *et al.*, "Language models are unsupervised multitask learners," *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.

[35] E. L. Campbell, R. Y. Mesía, L. Docío-Fernández, and C. García-Mateo, "Paralinguistic and linguistic fluency features for Alzheimer's disease detection," *Comput. Speech Lang.*, vol. 68, p. 101198, Jul. 2021.

[36] U. Sarawgi, W. Zulfikar, N. Soliman, and P. Maes, "Multimodal inductive transfer learning for detection of Alzheimer's dementia and its severity," 2020. [Online]. Available: https://arxiv.org/abs/2009.00700v1

[37] A. Pompili, T. Rolland, and A. Abad, "The INESC-ID multi-modal system for the ADReSS 2020 challenge," 2020. [Online]. Available: https://arxiv.org/abs/2005.14646

[38] M. S. S. Syed, Z. S. Syed, M. Lech, and E. Pirogova, "Automated screening for Alzheimer's dementia through spontaneous speech," in *Proc. Interspeech*, Oct. 2020, pp. 2222–2226.

[39] K. C. Fraser, J. A. Meltzer, and F. Rudzicz, "Linguistic features identify Alzheimer's disease in narrative speech," *J. Alzheimers Dis.*, vol. 49, pp. 407–422, 2015.

[40] M. Rohanian, J. Hough, and M. Purver, "Alzheimer's dementia recognition using acoustic, lexical, disfluency and speech pause features robust to noisy inputs," in *Proc. Interspeech*, Aug. 2021, pp. 3820–3824.

[41] G. Degottex, J. Kane, T. Drugman, T. Raitio, and S. Scherer, "COVAREP — a collaborative voice analysis repository for speech technologies," in *Proc. IEEE Int. Conf. Acoust, Speech Signal Process. (ICASSP)*, May 2014, pp. 960–964.

[42] J. Pennington, R. Socher, and C. Manning, "GloVe: Global vectors for word representation," in *Proc. Conf. Empir. Methods Nat. Lang. Process.*, Oct. 2014, pp. 1532–1543.

[43] M. Chatzianastasis, L. Ilias, D. Askounis, and M. Vazirgiannis, "Neural architecture search with multimodal fusion methods for diagnosing dementia," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, Jun. 2023, pp. 1–5.

[44] Z. Yu, J. Yu, C. Xiang, J. Fan, and D. Tao, "Beyond bilinear: Generalized multimodal factorized high-order pooling for visual question answering," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 12, pp. 5947–5959, Dec. 2018.

[45] B. Mirheidari, D. Blackburn, T. Walker, M. Reuber, and H. Christensen, "Dementia detection using automatic analysis of conversations," *Comput. Speech Lang.*, vol. 53, pp. 65–79, Jan. 2019.

[46] L. Ilias, D. Askounis, and J. Psarras, "Detecting dementia from speech and transcripts using transformers," *Comput. Speech Lang.*, vol. 79, p. 101485, Apr. 2023.

[47] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," 2020. [Online]. Available: https://arxiv.org/abs/2010.11929

[48] Y. Zhu, X. Liang, J. A. Batsis, and R. M. Roth, "Exploring deep transfer learning techniques for Alzheimer's dementia detection," *Front. Comput. Sci.-Switz*, vol. 3, p. 624683, May 2021.

[49] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient convolutional neural networks for mobile vision applications," 2017. [Online]. Available: https://arxiv.org/abs/1704.04861

[50] J. F. Gemmeke, D. P. W. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio set: An ontology and human-labeled dataset for audio events," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, Mar. 2017, pp. 776–780.

[51] A. T. Liu, S.-w. Yang, P.-H. Chi, P.-c. Hsu, and H.-y. Lee, "Mockingjay: Unsupervised speech representation learning with deep bidirectional transformer encoders," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, May 2020, pp. 6419–6423.

[52] R. B. Ammar and Y. B. Ayed, "Speech processing for early Alzheimer disease diagnosis: Machine learning based approach," in *Proc. IEEE/ACS Int. Conf. Comput. Syst. Appl. (AICCSA)*, Oct. 2018, pp. 1–8.

[53] J. Weiner and T. Schultz, "Selecting features for automatic screening for dementia based on speech," in *Proc. Lect. Notes Comput. Sci.*, Aug. 2018, pp. 747–756.

[54] S. H. Shum, N. Dehak, R. Dehak, and J. R. Glass, "Unsupervised methods for speaker diarization: An integrated and iterative approach," *IEEE Trans. Audio Speech Lang. Process.*, vol. 21, no. 10, pp. 2015–2028, Oct. 2013.

[55] J. Weiner, C. Frankenberg, J. Schröder, and T. Schultz, "Speech reveals future risk of developing dementia: Predictive dementia screening from biographic

interviews," in *Proc. IEEE Autom. Speech Recognit. Understanding Workshop (ASRU)*, Dec. 2019, pp. 674–681.

[56] T. Alhanai, R. Au, and J. Glass, "Spoken language biomarkers for detecting cognitive impairment," in *Proc. IEEE Autom. Speech Recognit. Understanding Workshop (ASRU)*, Dec. 2017, pp. 409–416.

[57] H. Zou and T. Hastie, "Regularization and variable selection via the elastic net," *J. R. Stat. Soc. Ser. B-Stat. Methodol.*, vol. 67, no. 2, pp. 301–320, Apr. 2005.

[58] Z. Shen and A. Chen, "Comprehensive relative importance analysis and its applications to high dimensional gene expression data analysis," *Knowledge-Based Syst.*, vol. 203, p. 106120, Sep. 2020.

[59] R. Tibshirani, "Regression shrinkage and selection via the Lasso," *J. R. Stat. Soc. Ser. B-Stat. Methodol.*, vol. 58, no. 1, pp. 267–288, Jan. 1996.

[60] Y. Qin, T. Lee, and A. P. H. Kong, "Automatic assessment of speech impairment in Cantonese-speaking people with aphasia," *IEEE J. Sel. Top. Signal Process.*, vol. 14, no. 2, pp. 331–345, Feb. 2020.

[61] L. Palmerini, L. Rocchi, S. Mellone, F. Valzania, and L. Chiari, "Feature selection for accelerometer-based posture analysis in Parkinson's disease," *IEEE T. Inf. Technol. Biomed.*, vol. 15, no. 3, pp. 481–490, May 2011.

[62] H. Peng, F. Long, and C. Ding, "Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 8, pp. 1226–1238, Aug. 2005.

[63] S. Wang, D. Li, X. Song, Y. Wei, and H. Li, “A feature selection method based on improved Fisher’s discriminant ratio for text sentiment classification,” *Expert Syst. Appl.*, vol. 38, no. 7, pp. 8696–8702, Jul. 2011.

[64] M.-W. Mak and S.-Y. Kung, “Fusion of feature selection methods for pairwise scoring SVM,” *Neurocomputing*, vol. 71, no. 16–18, pp. 3104–3113, Oct. 2008.

[65] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik, “Gene selection for cancer classification using support vector machines,” *Machine Learning*, vol. 46, no. 1/3, pp. 389–422, Jan. 2002.

[66] L. Meier, S. V. D. Geer, and P. Bühlmann, “The group Lasso for logistic regression,” *J. R. Stat. Soc. Ser. B-Stat. Methodol.*, vol. 70, no. 1, pp. 53–71, Jan. 2008.

[67] L. Jacob, G. Obozinski, and J.-P. Vert, “Group lasso with overlap and graph lasso,” in *Proc. ACM Int. Conf. Proc. Ser.*, Jun. 2009, pp. 433–440.

[68] J. Peng, J. Zhu, A. Bergamaschi, W. Han, D.-Y. Noh, J. R. Pollack, and P. Wang, “Regularized multivariate regression for identifying master predictors with application to integrative genomics study of breast cancer,” *Ann. Appl. Stat.*, vol. 4, no. 1, p. 53, Mar. 2010.

[69] L. Breiman, “Random forests,” *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, Oct. 2001.

[70] Y. Hechtlinger, “Interpretation of prediction models using the input gradient,” 2016. [Online]. Available: http://arxiv.org/abs/1611.07634

[71] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov,

"Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, Jan. 2014.

[72] Y. Gal, J. Hron, and A. Kendall, "Concrete dropout," in *Proc. Advances Neural Inf. Process. Syst. (NIPS)*, May 2017, pp. 3584–3593.

[73] T. Hastie, R. Tibshirani, J. H. Friedman, and J. H. Friedman, *The elements of statistical learning: data mining, inference, and prediction.* Springer, 2009, vol. 2.

[74] S. Luz, F. Haider, S. de la Fuente, D. Fromm, and B. MacWhinney, "Alzheimer's dementia recognition through spontaneous speech: The ADReSS challenge," 2020. [Online]. Available: https://arxiv.org/abs/2004.06833

[75] Y. Qin, W. Liu, Z. Peng, S.-I. Ng, J. Li, H. Hu, and T. Lee, "Exploiting pre-trained ASR models for Alzheimer's disease recognition through spontaneous speech," 2021. [Online]. Available: https://arxiv.org/abs/2110.01493

[76] J. Li, J. Yu, Z. Ye, S. Wong, M. W. Mak, B. Mak, X. Liu, and H. Meng, "A comparative study of acoustic and linguistic features classification for Alzheimer's disease detection," in *Proc. IEEE Int. Conf. Acoust, Speech Signal Process. (ICASSP)*, Jun. 2021, pp. 6423–6427.

[77] B. H. Davis and M. Maclagan, "Examining pauses in Alzheimer's discourse," *Am. J. Alzheimers Dis. Other Dement.*, vol. 24, no. 2, pp. 141–154, Apr. 2009.

[78] J. Yuan, M. Liberman *et al.*, "Speaker identification on the scotus corpus," *J. Acoust. Soc. Am.*, vol. 123, no. 5, p. 3878, May 2008.

[79] B. Schuller, S. Steidl, A. Batliner, F. Burkhardt, L. Devillers, C. Müller, and

S. Narayanan, “The INTERSPEECH 2010 paralinguistic challenge,” in *Proc. Interspeech*, Sep. 2010, pp. 2794–2797.

[80] J. Chen, M. Stern, M. J. Wainwright, and M. I. Jordan, “Kernel feature selection via conditional covariance minimization,” 2017. [Online]. Available: https://arxiv.org/abs/1707.01164

[81] L. Yang, W. Wei, S. Li, J. Li, and T. Shinozaki, “Augmented adversarial self-supervised learning for early-stage Alzheimer’s speech detection,” in *Proc. Interspeech*, Sep. 2022, pp. 541–545.

[82] S. Ahmed, A.-M. F. Haigh, C. A. de Jager, and P. Garrard, “Connected speech as a marker of disease progression in autopsy-proven Alzheimer’s disease,” *Brain*, vol. 136, no. 12, pp. 3727–3737, Oct. 2013.

[83] W. Jarrold, B. Peintner, D. Wilkins, D. Vergryi, C. Richey, M. L. Gorno-Tempini, and J. Ogar, “Aided diagnosis of dementia type through computer-based analysis of spontaneous speech,” in *Proc. Annu. Meet. Assoc. Comput. Linguist.*, Jun. 2014, pp. 27–37.

[84] S. Luz, “Longitudinal monitoring and detection of Alzheimer’s type dementia from spontaneous speech data,” in *Proc. IEEE Symp. Comput.-Based Med. Syst.*, Jun. 2017, pp. 45–46.

[85] K. M. Robinson, M. Grossman, T. White-Devine, and M. D'Esposito, “Category-specific difficulty naming with verbs in Alzheimer’s disease,” *Neurology*, vol. 47, no. 1, pp. 178–182, Jul. 1996.

[86] F. Haider, S. Pollak, P. Albert, and S. Luz, “Emotion recognition in low-resource settings: An evaluation of automatic feature selection methods,” *Comput. Speech Lang.*, vol. 65, p. 101119, Jan. 2021.

[87] C. Ambroise and G. J. McLachlan, “Selection bias in gene extraction on the basis of microarray gene-expression data,” *Proc. Natl. Acad. Sci. U. S. A.*, vol. 99, no. 10, pp. 6562–6566, Apr. 2002.

[88] S. S. Xu, M.-W. Mak, K. H. Wong, H. Meng, and C. Y. Kwok, “Speaker turn aware similarity scoring for diarization of speech-based cognitive assessments,” in *Proc. Asia-Pacific Signal Inf. Process. Assoc. Annu. Summit Conf. (APSIPA ASC)*, Dec. 2021, pp. 1299–1304.

[89] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, “PyTorch: An imperative style, high-performance deep learning library,” in *Proc. Adv. neural inf. proces. syst. (NIPS)*, Dec. 2019, pp. 8026–8037.

[90] K. He, X. Zhang, S. Ren, and J. Sun, “Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification,” in *Proc. IEEE Int. Conf. Comput. Vision (ICCV)*, Dec. 2015, pp. 1026–1034.

[91] Y. Huang, W. Jin, Z. Yu, and B. Li, “Supervised feature selection through deep neural networks with pairwise connected structure,” *Knowledge-Based Syst.*, vol. 204, p. 106202, Sep. 2020.

[92] K. H. Brodersen, C. S. Ong, K. E. Stephan, and J. M. Buhmann, “The balanced accuracy and its posterior distribution,” in *Proc. Int. Conf. Pattern Recognit. (ICPR)*, Aug. 2010, pp. 3121–3124.

[93] Z. S. Nasreddine, N. A. Phillips, V. Bédirian, S. Charbonneau, V. Whitehead, I. Collin, J. L. Cummings, and H. Chertkow, “The Montreal Cognitive Assess-

ment, MoCA: A brief screening tool for mild cognitive impairment," *J. Am. Geriatr. Soc.*, vol. 53, no. 4, pp. 695–699, Apr. 2005.

[94] K. Clark, M.-T. Luong, Q. V. Le, and C. D. Manning, "ELECTRA: Pre-training text encoders as discriminators rather than generators," 2020. [Online]. Available: https://arxiv.org/abs/2003.10555

[95] M. Komeili, C. Pou-Prom, D. Liaqat, K. C. Fraser, M. Yancheva, and F. Rudzicz, "Talk2Me: Automated linguistic data collection for personal assessment," *PLoS One*, vol. 14, no. 3, p. e0212342, Mar. 2019.

[96] M. Rohanian, J. Hough, and M. Purver, "Multi-modal fusion with gating using audio, lexical and disfluency features for Alzheimer's dementia recognition from spontaneous speech," 2021. [Online]. Available: https://arxiv.org/abs/2106.09668

[97] Z. S. Syed, S. A. Memon, M. S. Shah, and A. S. Syed, "Introducing the Urdu-Sindhi speech emotion corpus: A novel dataset of speech recordings for emotion recognition for two low-resource languages," *Int. J. Adv. Comput. Sci. Appl.*, vol. 11, no. 4, pp. 1–6, May 2020.

[98] Z. S. Syed, K. Sidorov, and D. Marshall, "Automated screening for bipolar disorder from audio/visual modalities," in *Proc. Audio/Vis. Emot. Chall. Workshop*, Oct. 2018, pp. 39–45.

[99] N. Wang, Y. Cao, S. Hao, Z. Shao, and K. Subbalakshmi, "Modular multimodal attention network for Alzheimer' disease detection using patient audio and language data," in *Proc. Interspeech*, Aug. 2021, pp. 3835–3839.

[100] W.-N. Hsu, B. Bolte, Y.-H. H. Tsai, K. Lakhotia, R. Salakhutdinov, and A. Mohamed, "HuBERT: Self-supervised speech representation learning

by masked prediction of hidden units," 2021. [Online]. Available: https://arxiv.org/abs/2106.07447

[101] S. Chen, C. Wang, Z. Chen, Y. Wu, S. Liu, Z. Chen, J. Li, N. Kanda, T. Yoshioka, X. Xiao, J. Wu, L. Zhou, S. Ren, Y. Qian, Y. Qian, J. Wu, M. Zeng, X. Yu, and F. Wei, "WavLM: Large-scale self-supervised pre-training for full stack speech processing," 2021. [Online]. Available: https://arxiv.org/abs/2110.13900

[102] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever, "Robust speech recognition via large-scale weak supervision," 2022. [Online]. Available: https://arxiv.org/abs/2212.04356

[103] X. Ke, M. W. Mak, J. Li, and H. M. Meng, "Dual dropout ranking of linguistic features for Alzheimer's disease recognition," in *Proc. Asia-Pacific Signal Inf. Process. Assoc. Annu. Summit Conf. (APSIPA ASC)*, Dec. 2021, pp. 743–749.

Appendix A

ALTERNATE LEARNING ALGORITHM OF DDR

Algorithm 1 Alternate learning algorithm of DDR.

```
Require: Operator network with parameters $\psi$ and selector network with parameters $\varphi$
Require: The size of dropout mask subset $|\mathcal{Z}|$, size of mini-batch $|\mathcal{M}|$, and number of training iterations $n$
Output: Dropout rates $\boldsymbol{\theta}_n$
Initialize dropout rates as $\boldsymbol{\theta}_0$
for $i \leftarrow 1$ to $n$ do
    Obtain a dropout mask subset $\mathcal{Z}$ with size $|\mathcal{Z}|$ using Eq. (3.1)
    for $j \leftarrow 1$ to $|\mathcal{Z}|$ do
        Compute the operator loss given $\boldsymbol{z}_i^{(j)}$:
            $$\ell_{O,i}^{(j)} = \frac{1}{|\mathcal{M}|} \sum_{(\boldsymbol{x},\boldsymbol{y}) \in \mathcal{M}} l(\boldsymbol{x} \odot \boldsymbol{z}_i^{(j)}, \boldsymbol{y}; \psi_i)$$
    end for
    Compute the average operator loss on $\mathcal{Z}$:
        $$\mathcal{L}_O\left(\mathcal{M}, \mathcal{Z}; \psi_i\right) = \frac{1}{|\mathcal{Z}|} \sum_{j=1}^{|\mathcal{Z}|} \ell_{O,i}^{(j)}$$
    Update operator network's parameters:
        $$\psi_i \leftarrow \psi_i - \eta \nabla_{\psi} \mathcal{L}_O\left(\mathcal{M}, \mathcal{Z}; \psi_i\right)\big|_{\psi=\psi_i}$$
    for $j \leftarrow 1$ to $|\mathcal{Z}|$ do
        Compute the selector loss given $\boldsymbol{z}_i^{(j)}$:
            $\ell_{S,i}^{(j)} = \left| f_S(\boldsymbol{z}_i^{(j)}; \varphi_i) - \ell_{O,i}^{(j)} \right| \Big/ \sum_{k=1}^{d} (1 - z_{i,k}^{(j)})$
    end for
    Compute the average selector loss on $\mathcal{Z}$:
        $\mathcal{L}_S\left(\mathcal{Z}(\boldsymbol{\theta}); \varphi_i\right) = \frac{1}{|\mathcal{Z}|} \sum_{j=1}^{|\mathcal{Z}|} \ell_{S,i}^{(j)};$
    Update selector network's parameters:
        $\varphi_i \leftarrow \varphi_i - \eta \nabla_{\varphi} \mathcal{L}_S\left(\mathcal{Z}(\boldsymbol{\theta}); \varphi_i\right)\big|_{\varphi=\varphi_i}$
    Update dropout rates:[1]
        $\boldsymbol{\theta}_i \leftarrow \boldsymbol{\theta}_i - \eta \sum_{j=1}^{|\mathcal{Z}|} \nabla_{\boldsymbol{z}(\boldsymbol{\theta})} \mathcal{L}_S\left(\mathcal{Z}(\boldsymbol{\theta}); \varphi_i\right) \nabla_{\boldsymbol{\theta}} \boldsymbol{z}(\boldsymbol{\theta})\big|_{\boldsymbol{\theta}=\boldsymbol{\theta_i}, \boldsymbol{z}=\boldsymbol{z}_i^{(j)}}$
end for
```

[1] The gradient is based on the chain rule: $\frac{\partial \mathcal{L}_S}{\partial \boldsymbol{\theta}} = \frac{\partial \mathcal{L}_S}{\partial \boldsymbol{z}} \cdot \frac{\partial \boldsymbol{z}}{\partial \boldsymbol{\theta}}$.

Appendix B

ALTERNATE LEARNING ALGORITHM OF DFR

Algorithm 2 Alternate learning algorithm of DFR.

```
Require: Operator network with parameters $\psi$ and selector network with parameters $\varphi$
Require: The size of feature mask subset $|\mathcal{Z}|$, size of mini-batch $|\mathcal{M}|$, number of selected features $s$, number of flipping $p$, and number of training iterations $n$
Ensure: Feature importance vector $\boldsymbol{c}^{(n)}$
Randomly initialize $\boldsymbol{z}_1^{(0)}$
for $i \leftarrow 1$ to $n$ do
    Compute $\boldsymbol{c}^{(i)}$ using Eq. (3.11) based on $\varphi^{(i)}$;
     generate the optimal feature mask vector $\boldsymbol{z}_2^{(i)}$ based on $\boldsymbol{c}^{(i)}$;
     generate $|\mathcal{Z}| - 2$ candidate feature mask vectors $\{\boldsymbol{z}_3^{(i)}, \dots, \boldsymbol{z}_{|\mathcal{Z}|}^{(i)}\}$;
     produce $\mathcal{Z} = \{\boldsymbol{z}_1^{(i-1)}, \boldsymbol{z}_2^{(i)}, \boldsymbol{z}_3^{(i)}, \dots, \boldsymbol{z}_{|\mathcal{Z}|}^{(i)}\}$
    for $j \leftarrow 1$ to $|\mathcal{Z}|$ do
        Compute the operator loss given $\boldsymbol{z}_j^{(i)}$:
            $$\ell_{O,j}^{(i)} = \frac{1}{|\mathcal{M}|} \sum_{(\boldsymbol{x},\boldsymbol{y}) \in \mathcal{M}} l\left(\boldsymbol{x} \odot \boldsymbol{z}_j^{(i)}, \boldsymbol{y}; \psi^{(i)}\right)$$
    end for
    Compute the average operator loss on $\mathcal{Z}$:
            $$\mathcal{L}_O\left(\mathcal{M}, \mathcal{Z}; \psi^{(i)}\right) = \frac{1}{|\mathcal{Z}|} \sum_{j=1}^{|\mathcal{Z}|} \ell_{O,j}^{(i)}$$
    Update operator network's parameters:
        $psi^{(i)} \leftarrow \psi^{(i)} - \eta \nabla_{\psi} \mathcal{L}_O \left( \mathcal{M}, \mathcal{Z}; \psi^{(i)} \right) \big|_{\psi = \psi^{(i)}}$
    **for** $j \leftarrow 1$ to $|\mathcal{Z}|$ **do**
        Compute the selector loss given $\boldsymbol{z}_j^{(i)}$:
            $\ell_{S,j}^{(i)} = \left| f_S(\boldsymbol{z}_j^{(i)}; \varphi^{(i)}) - \ell_{O,j}^{(i)} \right|$
    **end for**
    Compute the average selector loss on $\mathcal{Z}$:
        $\mathcal{L}_S \left( \mathcal{Z}; \varphi^{(i)} \right) = \frac{1}{|\mathcal{Z}|} \sum_{j=1}^{|\mathcal{Z}|} \ell_{S,j}^{(i)}$
    Update selector network's parameters:[1]
        $\varphi^{(i)} \leftarrow \varphi^{(i)} - \eta \nabla_{\varphi} \mathcal{L}_S \left( \mathcal{Z}; \varphi^{(i)} \right) \big|_{\varphi = \varphi^{(i)}}$
    Assign the best feature mask vector in $\mathcal{Z}$ to $\boldsymbol{z}_1^{(i)}$:
        $\boldsymbol{z}_1^{(i)} \leftarrow \arg\min_{j} \quad \ell_{O,j}^{(i)}$
**end for**
```

[1] Refer to [73] for the gradient of absolute error loss.

Appendix C

LIST OF LINGUISTIC/LEXICAL FEATURES

ADReSS dataset

Duration: total time of the sample

Total Utts: total number of utterances per minute

MLU Utts: number of utterances used to compute mean length of utterances (MLU)

MLU Words: the number of words/MLU Utts

MLU Morphemes: the number of morphemes/MLU Utts

FREQ types: total word types as counted by FREQ command

FREQ tokens: total word tokens as counted by FREQ command

FREQ TTR: type-token ratio

Words/min: words per minute (FREQ tokens/Duration converted to minutes)

Verbs/Utt: verbs per utterance

% **Word Errors:** percentage of words that are coded as errors [*]

Utt Errors: number of utterances coded as errors per minute

Density: measure of propositional idea density

% **Nouns:** percentage of nouns

% **Plurals:** percentage of plurals

% **Verbs:** percentage of verbs, including those tagged as verb, participle, and copula

% **Aux:** percentage of auxiliaries

% **3S:** percentage of third person singular

% **1S/3S:** percentage of identical forms for first and third person (e.g., I was, he was)

% **Past:** percentage of past tenses

% **PastP:** percentage of past participles

% **PresP:** percentage of present participles

% **prep:** percentage of present prepositions

% **adv:** percentage of adverbs

% **adj:** percentage of adjectives

% **conj:** percentage of conjunctions

% **det:** percentage of determiners

% **pro:** percentage of pronouns

noun/verb ratio: the ratio of nouns to verbs

open/closed ratio: the ratio of open class words to closed class words

# **open-class:** total number of open class words per minute

# **closed-class:** total number of closed class words per minute

# **retracing:** number of retracings (self-corrections or changes) per minute

# **repetition:** number of repetitions per minute

AD2021 dataset

# **Utts:** number of utterances per minute

Words/Utt: the mean number of words per utterance

Types: total word types

Tokens: total word tokens

TTR: type-token ratio

Average Word Freq: the average word frequency

Median Word Freq: the median word frequency

% **AD:** percentage of adverbs

% **AS:** percentage of aspect markers

% **BA:** percentage of ba-constructions

% **CC:** percentage of coordinating conjunctions

% **CD:** percentage of cardinal numbers

% **CS:** percentage of subordinating conjunctions

% **DT:** percentage of determiners

% **ETC:** percentage of words like “etc.”

% **FW:** percentage of foreign words

% **IC:** percentage of incomplete components

% **IJ:** percentage of interjections

% **JJ:** percentage of other noun-modifiers

% **LC:** percentage of localizers

% **M:** percentage of measure words

% **MSP:** percentage of other particles

% **NN:** percentage of common nouns

% **NR:** percentage of proper nouns

% **NT:** percentage of temporal nouns

% **OD:** percentage of ordinal numbers

% **PN:** percentage of pronouns

% **SP:** percentage of sentence final particles

% **VA:** percentage of predicative adjectives

% **VC:** percentage of be words

% **VE:** percentage of main verbs

% **VV:** percentage of other verbs

pronouns/nouns ratio: the ratio of pronouns to nouns

nouns/verbs ratio: the ratio of nouns to verbs

Max Tree Height: the maximum parsed tree height

Mean Tree Height: the mean parsed tree height

Median Tree Height: the median parsed tree height

JCCOCC-MoCA dataset

# **Utts:** number of utterances per minute

Words/Utt: the mean number of words per utterance

Types: total word types

Tokens: total word tokens

TTR: type-token ratio

% **Ag:** percentage of adjective morphemes

% **a:** percentage of adjectives

% **ad:** percentage of adjectives as adverbial

% **an:** percentage of adjectives with nominal function

% **Bg:** percentage of non-predicate adjective morphemes

% **b:** percentage of non-predicate adjectives

% **c:** percentage of conjunctions

% **Dg:** percentage of adverb morphemes

% **d:** percentage of adverbs

% **e:** percentage of interjections

% **f:** percentage of localizers

% **i:** percentage of idioms

% **m:** percentage of numerals

% **n:** percentage of common nouns

% **nr:** percentage of personal names

% **ns:** percentage of place names

% **o:** percentage of onomatopoeias

% **p:** percentage of prepositions

% **r:** percentage of pronouns

% **v:** percentage of verbs

pronouns/nouns ratio: the ratio of pronouns to nouns

nouns/verbs ratio: the ratio of nouns to verbs