MASSEY UNIVERSITY
TE KUNENGA KI PŪREHUROA
UNIVERSITY OF NEW ZEALAND

Graph Learning and Its Applications

A thesis presented in partial fulfilment of the

requirements for the degree of

Doctor of Philosophy

in

Computer Science

Massey University, Albany, Auckland,

New Zealand

Rongyao Hu

2022

Abstract

Since graph features consider the correlations between two data points to provide high-order information, *i.e.,* more complex correlations than the low-order information which considers the correlations in the individual data, they have attracted much attention in real applications. The key of graph feature extraction is the graph construction. Previous study has demonstrated that the quality of the graph usually determines the effectiveness of the graph feature. However, the graph is usually constructed from the original data which often contain noise and redundancy. To address the above issue, graph learning is designed to iteratively adjust the graph and model parameters so that improving the quality of the graph and outputting optimal model parameters. As a result, graph learning has become a very popular research topic in traditional machine learning and deep learning. Although previous graph learning methods have been applied in many fields by adding a graph regularization to the objective function, they still have some issues to be addressed.

This thesis focuses on the study of graph learning aiming to overcome the drawbacks in previous methods for different applications. We list the proposed methods as follows.

- We propose a traditional graph learning method under supervised learning to consider the robustness and the interpretability of graph learning. Specifically, we propose utilizing self-paced learning to assign important samples with large weights, conducting feature selection to remove redundant features, and learning a graph matrix from the low-dimensional data of the original data to preserve the local structure of the data. As a consequence, both important samples and useful features are used to select support vectors in the SVM framework.
- We propose a traditional graph learning method under semi-supervised learning to explore parameter-free fusion of graph learning. Specifically, we first employ the discrete wavelet transform and Pearson correlation coefficient to obtain multiple fully connected Functional Connectivity brain Networks (FCNs) for every subject, and then learn a sparsely connected FCN for every subject. Finally, the ℓ_1-SVM is employed to learn the important features and conduct disease diagnosis.
- We propose a deep graph learning method to consider graph fusion of graph learning. Specifically, we first employ the Simple Linear Iterative Clustering (SLIC) method to obtain multi-scale features for every image, and then design a new graph fusion method to fine-tune features of every scale. As a result, the multi-scale feature fine-tuning, graph learning, and feature learning are embedded into a unified framework.

All proposed methods are evaluated on real-world data sets, by comparing to state-of-the-art methods. Experimental results demonstrate that our methods outperformed all comparison methods.

In memory of my mother

Acknowledgements

I would like to express my gratitude to everyone who has helped me during my Doctoral study and anyone who has accompanied me on my Doctoral study to complete this qualification.

I owe special gratitude to my chief supervisor Associate Professor Xiaofeng Zhu, who have paid attention to me and guided me during my Doctoral study. His wealth of research experience and rigour attitude have deeply influenced me. He also acted as a beacon of light to guide me in my study and life. For my study, he taught me that I need to be passionate about my research work. The more challenges I face, the more time I can be trained and the skills will be improved. The attitude of leading by example always inspires me that I should never give up on the road of life. For my life, he taught me the principles of life in the world and keeping exercising for our bodies healthy, which are the benchmark for all work in my life. I will always keep these guidelines in my mind as I move forward. I will forever preserve thanks for that.

I also show my respect and sincere thanks to my co-supervisors Dr. Teo Susnjak and Dr. Tong Liu for their support on my research, showing me the knowledge and invaluable guidance. I was so lucky to have teachers like you at Massey University.

I would like to express my thanks to the friends whom I have made in the School of Natural and Computational Sciences and the members of our research team for their companionship and assistance. I will always remember the good times we had.

I would like to gratefully acknowledge the School of Natural and Computational Sciences at Massey University for supporting my Doctoral study and providing me with the financial support. Meanwhile, I also would like to thank Professor Dianner Brunton and Associate Professor Alona Ben-Tal for the development of the School, and to Ms. Annette Warbrooke and Ms. Linh Mill for their help and support.

I would like to express my deep appreciation to my parents for their support and unconditional love. I am very grateful to them for supporting me to start my Doctoral study. Lastly, I would like to thank myself who is not a smart and clever student, as it is not being easy to complete this study.

Publications

- **R. Hu**, X. Zhu, Y. Zhu, and J. Gan. Robust SVM with adaptive graph learning. *World Wide Web*, 23(3): 1945-1968, 2020. (Refer to Chapter 3)
- **R. Hu**, Z. Peng, X. Zhu, J. Gan, Y. Zhu, J. Ma, and G. Wu. Multi-Band Brain Network Analysis for Functional Neuroimaging Biomarker Identification. *IEEE Transactions on Medical Imaging*, 40(12): 3843-3855, 2021. (Refer to Chapter 4)
- **R. Hu**, Z. Deng, and X. Zhu. Multi-scale Graph Fusion for Co-saliency Detection. In *AAAI*, pp. 1-9, 2021. (Refer to Chapter 5)
- **R. Hu**, J. Gan, X. Zhu, T. Liu, and X. Shi. Multi-task Multi-modality SVM for Early COVID-19 Diagnosis using Chest CT Data. *Information Processing & Management*, 59(1):102782, 2022.
- **R. Hu**, L. Zhang, and J. Wei. Adaptive Laplacian Support Vector Machine for Semi-supervised Learning. *The Computer Journal*, 64(7):1005-1015, 2021.
- **R. Hu**, and X. Zhu. Multi-model Graph Neural Networks for Functional Neuroimaging Identification. Submitted to *ACM Multimedia*, 2022.
- C. Yuan, Z. Zhong, C. Lei, X. Zhu, and **R. Hu***. Adaptive Reverse Graph Learning for Robust Subspace Learning. *Information Processing & Management*, No. 6, pp. 102733, 2021.
- L. Peng, **R. Hu**, F. Kong, J. Gan, Y. Mo, X. Shi, X. Zhu. Reverse Graph Learning for Graph Neural Network. *IEEE Transactions on Neural Networks and Learning Systems*, DOI: 10.1109/TNNLS.2022.3161030, 2022.
- J. Gan, Z. Peng, X. Zhu, **R. Hu**, J. Ma, and G. Wu. Brain functional connectivity analysis based on multi-graph fusion. *Medical Image Analysis*, pp. 102057, 2021.
- X. Zhu, B. Song, F. Shi, Y. Chen, **R. Hu**, J. Gan, W. Zhang, M. Li, L. Wang, Y. Gao, F. Shan, and D. Shen. Joint prediction and time estimation of COVID-19 developing severe symptoms using chest CT scan. *Medical Image Analysis*, Vol. 67, pp. 101824, 2021.
- J. Gan, X. Zhu, **R. Hu**, Y. Zhu, J. Ma, and Z. Peng. Multi-graph Fusion for Functional Neuroimaging Biomarker Detection, In *IJCAI*, pp. 580-586, 2020.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Artificial intelligence (AI) makes machines first learn experience (*i.e.,* the model) from training data and then apply new inputs (*i.e.,* testing data) in the learned experience to conduct human-like tasks. In these years, AI has been used for different kinds of real-world applications [45, 54], *e.g.,* electronic commerce, education, healthcare, *etc.* Different applications usually correspond to different tasks of AI. In general, the basic tasks of AI have classification, clustering, regression, and so on [154]. For example, the classification task first trains the model on a set of labelled transaction data, and then predicts if given unknown transactions are credit card frauds. Retail companies conduct the clustering task on the information (such as household income and household size) to identify the household groups which are similar to each other.

In order to solve applications by AI tasks, AI methods usually involve three steps, *i.e.,* feature learning, model construction, and model evaluation. To be specific, feature learning aims to learn informative and computer-recognisable features to provide mathematically and computationally convenient for model construction and model evaluation [26, 116]. Model construction focuses on learning a set of parameters related to the specific task to fit objective functions for individual tasks. Model evaluation analyzes the learned features and models by various evaluation metrics. In general, feature learning is the key step for AI methods [166, 220].

To make AI data available to solve the application issues, different feature learning methods are designed to have various feature forms include original features, graph features and deep features. Original features are extracted from the low-level characteristics (*e.g.,* edges and blobs of images) by color/texture [38], GIST descriptors [75], Local Binary Pattern (LBP) [56], Histogram of Oriented Gradient (HOG), and so on [69]. The original feature does not take into account the correlation between two data samples so that it often achieves worse model performance, compared to either graph features and deep features in many real-world applications. Graph features are extracted by taking into account the correlation between two samples, and they have been demonstrated to be more discriminative than original features [51]. Deep fea-

tures are iteratively obtained by performing the final tasks in deep models, but they need a large number of data to fit deep models. Recently, with the success of deep learning, deep features are becoming increasingly popular. Graph features have attracted more and more attentions in real-world applications since they can assist either original features or deep features in model construction and can be combined with these two kinds of features to learn more representative features [83, 128].

1.2 Motivations

The key of graph feature extraction is graph construction. The graph construction is to represent every node with the original feature of the sample and set the edges as the similarity (*i.e.,* correlation) between two nodes. Popular methods of the graph construction include the fully connected graph, the k-nearest neighbor (kNN) graph, the ε-nearest neighbor graph, *etc.* The fully connected graph connects every node to all nodes. The kNN graph connects every node to k nodes with the maximal similarity while the ε-nearest neighbor graph uses the threshold (*i.e.,* ε) to decide the connection number for every node. Obviously, the fully connected graph preserves the global structure of the data. By contrast, both the kNN graph and the ε-nearest neighbor graph preserve the local structure of the data by connecting every node with a subset of all nodes only. Weinberger *et al.* demonstrate that the local structure preservation is more important than the global structure preservation in many real-world applications such as manifold learning [165]. Moreover, the local structure preservation of graph features make it very useful for all kinds of applications. For example, the local structure preservation makes graph features consider the high-order information (*i.e.,* the correlation between two samples), while original features only take into account the low-order information, *i.e.,* the individual characteristics of the samples. In particular, embedding graph features with deep models is available to output deep features that are discriminative as well as preserve the local structure of the data.

Graph construction outputs a graph storing the similarity between any two data. Obviously, the quality of the graph directly influences the effectiveness of the graph feature. Many graphs are constructed from the original data, which often contain noise and redundancy. Hence, the quality cannot be guaranteed. To address this issue, a popular method is to adaptively update the graph and the model parameters in a unified framework. Specifically, the graph can be adjusted based on the update of the model parameters, called graph learning. Moreover, the model parameters are further adjusted by the optimized graph. As a result, the quality of the graph is improved and the model parameters are optimized. Hence, graph learning has become a very hot research topic in traditional machine learning and deep learning.

Previous graph learning methods include traditional graph learning and deep graph learning. Traditional graph learning aims to learn the graph feature by embedding graph learning with traditional machine learning models. To do this, it usually adds a graph regularization to the loss function. For example, Kang *et al.* propose to conduct graph learning by a low-rank regularization to the classification loss function [84]. Deep graph learning focuses on extracting deep features by integrating the graph constraint with the loss function in a unified framework. For

example, Chen *et al.* propose to iteratively learn graph features and deep features, so that both of them can be updated by each other [24]. In a word, graph learning in either traditional methods or deep methods are designed by adding a graph regularization into the loss function. However, deep graph learning often achieves better performance than traditional graph learning with the help of large number of training data. In real-world applications, it is difficult to always obtain a large number of training data. As a result, deep graph learning may be worse than traditional graph learning due to the over-fitting issue.

Although graph learning has been applied in many fields, it still has many issues to be addressed.

- Robustness and interpretability of traditional graph learning. The quality of original data influences the graph construction. Besides directly removing noisy samples and redundant features, the sample importance and the feature importance are possible solutions to improve the quality of the original data, and thus achieving robust graph learning. On the other hand, interpretability is very important in real-world applications. For example, the clinicians prefer to obtain effective and interpretable graph learning. Hence, it is highly desirable to conduct graph learning while taking into account the interpretability.
- Parameter-free fusion of traditional graph learning. Different graphs result in different graph features. These features are usually heterogeneous and are in different feature spaces, and thus they are not comparable. Hence, it is straightforward to smooth all graph features so that they are homogeneous in the common feature space.
- Graph fusion of deep graph learning. Co-saliency detection focuses on simulating the human visual system to perceive the scene for searching the common and salient regions from a group of images. To achieve this, previous methods usually extract multi-scale features to comprehensively characterize the images since multi-scale features can detect different patterns with different sizes. It is highly demand to investigate a fusion method to combine multiple dynamic Graph Convolutional Network (GCN) models to explore the common information among multi-scale features and the complementary information in individual features.

1.3 Contributions

This thesis aims to investigate different graph learning methods for different applications by overcoming the issues of previous graph learning methods. The main contributions of this thesis are summarized as follows:

- Objective 1: To solve the issues of the robustness and the interpretability of graph learning in a unified supervised framework, by using important samples and useful features to select support vectors in the Support Vector Machine (SVM) framework.
- Objective 2: To fuse the multi-scale features for functional neuroimaging biomarker identification by conducting joint graph feature learning and personalized disease diagnosis in semi-supervised learning.

- Objective 3: To smooth multiple graph features for co-salience detection by proposing a graph fusion of deep graph learning in semi-supervised learning.

1.4 Thesis structures

The organization of this thesis is listed as follows.

Chapter 2 proposes a comprehensive survey on the studies of previous graph learning methods, including traditional graph learning methods and deep graph learning methods.

Chapter 3 proposes a supervised graph learning method to meet Objective 1. The proposed method outperforms state-of-the-art methods in terms of binary classification tasks and multi-class classification tasks.

Chapter 4 proposes a semi-supervised graph learning method to meet Objective 2. The proposed method outperforms both traditional graph learning methods and deep graph learning methods in terms of disease diagnosis tasks on resting state functional Magnetic Resonance Imaging (fMRI) data.

Chapter 5 proposes a deep graph learning method to meet Objective 3. The proposed method outperforms both traditional methods and deep methods in terms of co-saliency detection tasks.

Chapter 6 concludes this thesis and discusses our future works.

Chapter 2

Literature review

2.1 Graph construction revisited

Graphs, also known as networks, can be found in different kinds of real applications, including traffic network, molecular protein graph, social media network, *etc*. Moreover, the graph contains the features of the samples and the local structure (*i.e.,* the correlations) among the data, so it provides a new way of data representation. As a result, the study on the graph (especially graph learning) has attracted much attention in data mining and machine learning [172, 207].

2.1.1 Similarity measurements

Graph construction is the key of graph learning, which needs to define the similarity measurement. Popular similarity measurement methods include parameter-free methods and parameter methods. Parameter-free methods calculate the similarity between two samples by employing different distance measurements, such as Euclidean distance, Mahalanobis distance, correlation methods, *etc*. For example, Mahalanobis distance calculates the distance between two samples by

$$(\mathbf{x}_i - \mathbf{x}_j)^T \Sigma^{-1} (\mathbf{x}_i - \mathbf{x}_j), \tag{2.1}$$

where $\mathbf{x}_i$ represents the feature of the i-th sample.

Cosine similarity calculate the cosine value between $\mathbf{x}_i$ and $\mathbf{x}_j$ to obtain the cosine distance by

$$(\mathbf{x}_i^T \mathbf{x}_j) / (\sqrt{\mathbf{x}_i^T \mathbf{x}_i} \sqrt{\mathbf{x}_j^T \mathbf{x}_j}). \tag{2.2}$$

Pearson correlation coefficient is defined as

$$((\mathbf{x}_i - \hat{\mathbf{x}}_i)^T (\mathbf{x}_j - \hat{\mathbf{x}}_j)) / (\sqrt{(\mathbf{x}_i - \hat{\mathbf{x}}_i)^T (\mathbf{x}_i - \hat{\mathbf{x}}_i)} \sqrt{(\mathbf{x}_j - \hat{\mathbf{x}}_j)^T (\mathbf{x}_j - \hat{\mathbf{x}}_j)}), \tag{2.3}$$

where $\hat{\mathbf{x}}_i$ denote the average value of $\mathbf{x}_i$.

Parameter methods adjust the similarity between two samples by the parameters. For example, Minkowski metric uses the parameter p to obtain the similarity as

$$(\mathbf{x}_i - \mathbf{x}_{jp})^{1/p}, \tag{2.4}$$

where $p > 0$.

Gaussian kernel function utilizes the parameter σ to adjust the similarity as

$$exp((-||\mathbf{x}_i - \mathbf{x}_j||_2^2)/2\sigma^2), \tag{2.5}$$

where σ is a kernel width.

2.1.2 Graph construction

Given the similarity measurements, the graph construction usually forms two kinds of graphs, *i.e.,* normal graphs and hypergraphs. Every edge in the normal graph connects two nodes only, whereas Every edge in the hypergraph connects at least two nodes.

2.1.2.1 Normal graph

The normal graph includes fully connected graphs and sparsely connected graphs. The fully connected graph connects every node to all nodes. It is easy to construct the fully connected graph which preserves the global structure of the data. However, it does not meet the requirements of real applications as the full connection makes the optimization of the objective function difficult. Sparsely connected graph connects every nodes by a part of all nodes so that it easily preserves the local structure among the data. As a result, sparsely connected graph has been demonstrated to be more convenient in real applications. Popular methods of sparsely connected graphs include k-nearest neighbor (kNN) graph, ε-nearest neighbor (ε-NN) graph, *etc.* The kNN graph connects every nodes with its top k nearest neighbors and ε-NN graph connects every nodes with the nodes within the similarity of threshold ε. We visualize the difference between the kNN graph and the ε-NN graph in Figure 2.1 and list their construction as follows.

The kNN graph employs Gaussian kernel function (*i.e.,* the parameter method) to measure the similarity between two nodes. Specifically, the similarity between the i-th node $\mathbf{x}_i$ and the j-th node $\mathbf{x}_j$ is defined as

$$s_{i,j} = \begin{cases} exp^{-\frac{||\mathbf{x}_i - \mathbf{x}_j||_2^2}{2\sigma^2}}, & if \ \ \mathbf{x}_i \in \mathcal{N}(\mathbf{x}_j), \\ 0, & otherwise, \end{cases} \tag{2.6}$$

where $\mathcal{N}(\mathbf{x}_j)$ denotes the set of k-nearest neighbors of the j-th node.

The ε-NN graph defines the similarity between two nodes by

$$s_{i,j} = \begin{cases} exp^{-\frac{||\mathbf{x}_i - \mathbf{x}_j||_2^2}{2\sigma^2}}, & if \ \ ||\mathbf{x}_i - \mathbf{x}_j||_2^2 \leq \varepsilon, \\ 0, & otherwise, \end{cases} \tag{2.7}$$

where ε denotes the threshold between the i-th node $\mathbf{x}_i$ and the j-th node $\mathbf{x}_j$.

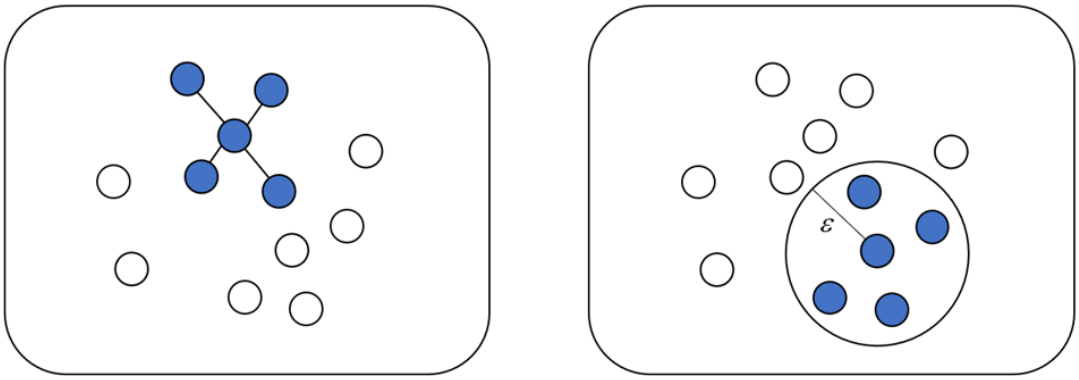
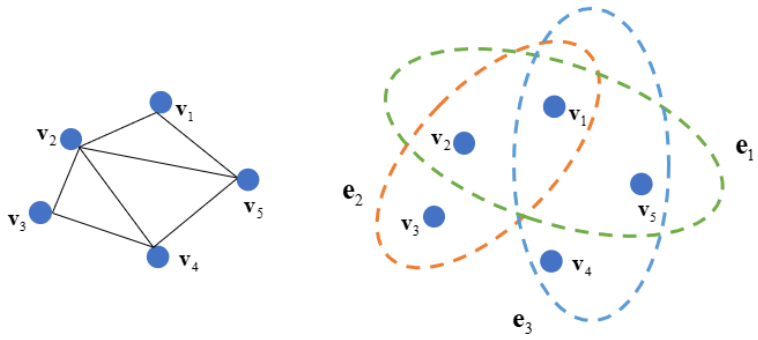


Figure 2.1: Visualization of the difference between the kNN graph and the ε-NN graph.



	$\mathbf{e}_1$	$\mathbf{e}_2$	$\mathbf{e}_3$
$\mathbf{v}_1$	1	1	1
$\mathbf{v}_2$	1	1	0
$\mathbf{v}_3$	0	1	0
$\mathbf{v}_4$	0	0	1
$\mathbf{v}_5$	1	0	1

Figure 2.2: Visualization of the difference between the normal graph and the hypergraph, where (a) is the normal graph, (b) is a hypergraph, and (c) is the incidence matrix in the hypergraph.

2.1.2.2 Hypergraph

The correlation among the data is more complex than the case where the edge connects only two nodes, so hypergraphs with every edge connecting more than two nodes have been proposed and applied in many applications [175, 213]. We visualize the difference between the normal graph and the hypergraph in Figure 2.2 and list the details of the hypergraph construction as follows.

Specifically, a hypergraph $\mathbf{G} = (\mathbf{V}, \mathbf{E}, \mathbf{W})$ consists of three parts, *i.e.,* a vertex set $\mathbf{V}$, a hyperedge set $\mathbf{E}$ and a hyperedge weight matrix $\mathbf{W}$. The weight of hyperedge e is defined as $w(e)$. The link correlation of the hypergraph $\mathbf{G}$ is expressed as an incidence matrix $\mathbf{H} \in \mathbb{R}^{|V|\times|E|}$, which is defined as follows:

$$\mathbf{H}(v,e) = \begin{cases} 1, & if \ \ v \in e, \\ 0, & otherwise. \end{cases} \tag{2.8}$$

According to the definition of the hypergraph, the degree matrix of hypergraph vertices and

hyperedges can be further expressed as

$$\begin{aligned} d(v) &= \sum_{e \in \mathbf{E}} w(e) h(v,e), \\ \delta(e) &= \sum_{v \in \mathbf{V}} h(v,e). \end{aligned} \tag{2.9}$$

In addition, two diagonal matrices $\mathbf{D}_v$ and $\mathbf{D}_e$ are defined to represent the pair angle matrices of hypergraph vertices and hyperedges, respectively. Based on the spectral graph theory in [213], the adjacency matrix $\mathbf{S}$ of the hypergraph is defined as

$$\mathbf{S} = \mathbf{H}\mathbf{W}\mathbf{H}^T - \mathbf{D}_v. \tag{2.10}$$

We use the normalized Laplacian matrix to establish the hypergraph Laplacian matrix by

$$\mathbf{L}_H = \mathbf{I} - \mathbf{D}_v^{-\frac{1}{2}} \mathbf{H}\mathbf{W}\mathbf{D}_e^{-1}\mathbf{H}^T\mathbf{D}_v^{-\frac{1}{2}}. \tag{2.11}$$

After constructing the graph, we use the following objective function to conduct graph learning,

$$\mathcal{L} = \mathcal{L}oss(\cdot) + \lambda \mathcal{R}eg(\cdot), \tag{2.12}$$

where $\mathcal{L}oss(\cdot)$ is the loss function and $\mathcal{R}eg(\cdot)$ is the graph regularization. Based on Eq. (2.12), both the graph and model parameters can be iteratively updated by each other until both of them reach the optimal solution.

In this thesis, previous graph learning methods can be summarized into traditional graph learning methods and deep graph learning methods.

2.2 Traditional graph learning

Graph learning enhances the model performance by capturing the geometrical information among samples [95, 194]. Based on the graph variations in the model construction, traditional graph learning methods include static graph learning and dynamic graph learning.

2.2.1 Static graph learning

Static graph learning [219] first learns the graph from the original data, and then fixes the graph throughout the whole optimization. The corresponding objective function can be expressed as

$$\mathcal{L}(\mathbf{W}) = \mathcal{L}oss(\mathbf{W}) + tr(\mathbf{X}^T\mathbf{L}\mathbf{X}), \tag{2.13}$$

where $\mathcal{L}(\mathbf{W})$ is the loss function with model parameters $\mathbf{W}$ and $tr(\mathbf{X}^T\mathbf{L}\mathbf{X})$ is the graph regularization term. $\mathbf{L} = (\mathbf{D} - \mathbf{S}) \in \mathbb{R}^{n \times n}$ is the Laplacian matrix, where $\mathbf{D} \in \mathbb{R}^{n \times n}$ represents a diagonal matrix. Eq. (2.13) indicates that the model parameters are learnt based on the graph constraints. That is, the outputs of Eq. (2.13) should preserve the local structure of the data.

In real applications, the original data usually contain redundancy and noise, and thus the quality of the graph will be influenced. To address this issue, Eq. (2.13) is used to preserve the local structure in the low-dimensional feature space, where redundancy and noise are removed as much as possible, *i.e.,*

$$\mathcal{L}(\mathbf{W}, \mathbf{\Theta}) = \mathcal{L}oss(\mathbf{W}) + tr(\mathbf{\Theta}^T \mathbf{X}^T \mathbf{L} \mathbf{X} \mathbf{\Theta}), \tag{2.14}$$

where $\mathbf{\Theta} \in \mathbb{R}^{d \times c}$ is the projection matrix.

Redundancy and noise in high-dimensional data make them ineffectively construct the model with the traditional machine learning. Based on the observation that redundancy and noise may lead to the low rank of the coefficient matrix of the data, reducing the rank of the matrix can effectively reduce the influence of redundancy and noise [111]. Therefore, a low-rank model for graph learning can be expressed as:

$$\mathcal{L}(\mathbf{W}, \mathbf{\Theta}) = \mathcal{L}oss(\mathbf{W}) + tr\left(\mathbf{\Theta}^T \mathbf{X}^T \mathbf{L} \mathbf{X} \mathbf{\Theta}\right) + rank(\mathbf{\Theta}), s.t., rank(\mathbf{\Theta}) < min(d, c), \tag{2.15}$$

where $rank(\mathbf{\Theta})$ is a low-rank regularization. d and c, respectively, indicate the number of the dimensionality and the class number.

While there are a large number of redundant features in the high-dimensional data, the ℓ_1-norm regularization is a new way for reducing the influence of redundant features, *i.e.,* feature selection. For example, MCFS [13] employs the ℓ_1-norm regularization to conduct feature selection by removing the influence of redundant features. Although the ℓ_1-norm regularization can reduce the influence of the outlier to some extent, it does not have the rotation invariance characteristics. Moreover, the ℓ_1-norm regularization can not describe the geometric structure of the data, so that it makes feature selection inaccurate. Hence, Ding *et al.* [29] replaces the ℓ_1-norm regularization with the square of the original ℓ_2-norm regularization to calculate the reconstruction error of the samples. As a result, it is not only robust to noisy data and outlier, but also can well preserve the geometric structure of the data with the rotation invariance.

2.2.2 Dynamic graph learning

Both Eq. (2.13) and Eq. (2.15) (*i.e.,* static graph learning) only update the model parameters with the unchanged graph. That is, graph learning is independent on the model construction. In particular, static graph learning is sensitive to either noise or redundancy. Therefore, to address the above issue, dynamic graph learning aims to jointly conduct model parameters and the graph. Compared to static graph learning, dynamic graph learning [98, 210, 217, 221, 222] iteratively update the graph in each iteration and the model parameters to reach their individually optimal solutions. For example, Zheng *et al.* [210] combine graph learning with feature selection in a unified framework to dynamically learn the graph from a low dimensional space and conduct feature selection. The objective function is defined as follows.

$$\begin{aligned} &\mathcal{L}(\mathbf{W}, \mathbf{\Theta}) = \mathcal{L}oss(\mathbf{W}) + \alpha \left\| x^i \mathbf{\Theta} - x^i \mathbf{\Theta} \right\|_2^2 s_{ij} + \beta \left\| \mathbf{S} \right\|_F^2 \\ &s.t., \forall i,\ s_i^T \mathbf{1} = 1, s_{ii} = 0, s_{ij} \geq 0,\ \ if\ \ i \in N(j),\ \ otherwise\ \ 0, \end{aligned} \tag{2.16}$$

where α and β are the parameters. Eq. (2.16) iteratively updates the graph $\mathbf{S}$ and the parameter matrix $\mathbf{W}$ until both of them reach convergence.

Recently, more and more studies focus on dynamic graph learning. For example, Zhu *et al.* consider dynamically adjust the similarity among the data and further utilize the nodes for the graph update [218]. Wu *et al.* construct the graph by using partial absorption random walk [171]. Wang *et al.* propose a joint semi-supervised learning for graph learning and label propagation [163]. Lei *et al.* extend [210] by considering both the low-rank constrain and subspace learning [98].

2.3 Deep graph learning

In recent years, deep learning methods on graphs (*i.e.,* deep graph learning for short) have gradually attracted attentions. Different from traditional graph learning methods, deep graph learning methods first define neural networks on the graph, and then adaptively learn node features and model parameters of the neural network. Among them, Graph Convolutional Networks (GCN) is a frontier research topic for deep graph learning.

GCN generalizes traditional deep learning techniques on the graph structured data. It mainly follows the message-passing framework, which consists of two main steps, *i.e.,* collecting messages from node neighbors which is the process of propagating information from neighbors to nodes, and passing messages which uses neural networks to update the node representation which is the process of aggregation of neighbor (structural) information. Existing methods can be partitioned into two groups, *i.e.,* statically deep graph learning and dynamically deep graph learning.

2.3.1 Statically deep graph learning

GCN is a well-known statically deep graph learning method, where the graph is used to preserve the local structure of the data during the process of representation learning. Specifically, given an initial graph $\mathbf{S} \in \mathbb{R}^{n\times n}$ storing the graph structure of the feature matrix $\mathbf{X} \in \{\mathbf{x}_1, ..., \mathbf{x}_n\} \in \mathbb{R}^{n\times d}$ where n and d, respectively, indicate the number of the samples and the features, the GCN taking both $\mathbf{S}$ and $\mathbf{X}$ as the inputs passes several hidden layers and one fully connected layer to output the new representation. More specifically, the representation learned by the m-th layer of the GCN can be obtained by:

$$\mathbf{H}^{(l+1)} = \rho(\mathbf{D}^{-\frac{1}{2}}\mathbf{S}\mathbf{D}^{-\frac{1}{2}}\mathbf{H}^{(l)}\mathbf{W}^{(l)}), \tag{2.17}$$

where $\mathbf{H}^{(l)} \in \mathbb{R}^{n\times d_l}$ denotes the output representation in the l-th hidden layer, $\mathbf{D} \in \mathbb{R}^{n\times n}$ is the diagonal matrix of $\mathbf{S}$ where d_{ii} is the summation of all elements in the i-th column of $\mathbf{S}$, $\mathbf{W}^{(l)} \in \mathbb{R}^{d_{l-1}\times d_l}$ is the weight matrix which needs to be trained in the l-th layer, d_l is the number of hidden units in the l-th layer, and $\rho(\cdot)$ is the activation function.

GCN has achieved great success on semi-supervised classification. However, previous methods only consider nodes that are a few propagation steps. Moreover, the size of the neighborhood in previous methods is hard to be extended. To deal with these issues, Klicpera *et al.* use the

correlation between the GCN and the PageRank to obtain an improved propagation scheme based on the personalized PageRank [88].

In real applications, either the over-smoothing issue or the over-fitting issue influences the effectiveness of deep graph learning. To deal with these two issues, Rong *et al.* suggest to randomly remove some edges from $\mathbf{S}$ to reduce the convergence speed of the over-smoothing issue [132]. Chen *et al.* enable GCN to express a K order polynomial filter with arbitrary coefficients. As a result, the issue of over-fitting is solved and it takes advantage of the powerful nonlinear capabilities of deep networks [20].

GCNs obtain inspiration from recent deep learning methods, but they may inherit unnecessary complexity and redundant computation. To deal with the issues, Wu *et al.* propose a simplifying GCN to reduce the complexity through successively removing non-linearities and collapsing weight matrices between consecutive layers [169].

Many variants of the GCN have been proposed. For example, James *et al.* propose a diffusion convolutional neural network to consider the difference in the importance of links between two nodes [5]. Hamilton *et al.* propose to learn a function that generates embeddings by sampling and aggregating features from the local neighborhood of the node [59]. Chen *et al.* propose a fast learning method with GCN via importance sampling [18]. Huang *et al.* propose to promote the message passing over distant nodes by applying skip connections, and thus accelerating the training process of GCNs [70]. Jin *et al.* propose to effectively and efficiently preserve node similarity while exploiting the graph structure [80].

2.3.2 Dynamically deep graph learning

Different from statically deep graph learning, dynamically deep graph learning usually learns node connections and edge weights automatically from the data. In addition, it can also be optimized in the same framework with related learning tasks, so as to output an optimal graph representation suitable for subsequent tasks. To some extent, dynamically deep graph learning solves the issues of statically deep graph learning, and has been widely used in machine learning.

Many techniques of dynamically deep graph learning have been introduced. For example, Jiang *et al.* employ the graph learning technique in traditional machine learning to add one more regularization into the original GCN framework [78], *i.e.,*

$$\begin{aligned} \mathcal{L}_{GL} : \min_{\mathbf{S}} \sum_{i,j=1}^{n} \|\mathbf{x}_i - \mathbf{x}_j\|_F^2 s_{ij} + \lambda_1 \|\mathbf{S}\|_F^2 \\ s.t., \forall i, \mathbf{s}_i \mathbf{1} = 1, s_{ij} \geq 0. \end{aligned} \tag{2.18}$$

Based on the proposed graph learning regularization, this method updates $\mathbf{S}$ via a single-layer neural network, which is parameterized by a weight vector $\mathbf{a} = (a_1, a_2, ..., a_p)^T \in \mathbb{R}^{p \times 1}$, and each element of $\mathbf{S}$ is defined as

$$s_{i,j} = \frac{exp(LeakyReLU(\mathbf{a}^T||\mathbf{x}_i - \mathbf{x}_j||))}{\sum_{j=1}^{n} exp(LeakyReLU(\mathbf{a}^T||\mathbf{x}_i - \mathbf{x}_j||))}. \tag{2.19}$$

Following [78], Ji *et al.* propose to introduce a self-expressive layer between the encoder and the decoder to mimic the self-expressiveness feature [76]. To encode self-expressiveness, Ji *et al.* introduce a regularization as follows

$$\|\mathbf{X} - \mathbf{XS}\| + \lambda\|\mathbf{S}\|_p, \tag{2.20}$$

where $\mathbf{S}$ is the a trainable parameter.

To further improve the robustness of dynamically deep graph learning, Jin *et al.* propose to simultaneously learn the clean graph structure from perturbed graph and GCN parameters to defend against adversarial attacks [81]. Specifically, this method exploring the low rank and sparsity properties of the graph is defined as follows,

$$\|\mathbf{S} - \mathbf{A}\|_F^2 + \lambda_1\|\mathbf{A}\|_1 + \lambda_2\|\mathbf{A}\|_*, \tag{2.21}$$

where $\mathbf{S}$ is the original graph and $\mathbf{A}$ is the updated graph.

Most recently, Fu *et al.* propose to first constantly optimize the high-order structural correlations between data points and then fit the local geometry information of the data exactly for semi-supervised classification [40]. Jiang *et al.* design to conduct graph convolutional learning and labeling on both the inter-graph and the intra-graph for co-saliency estimation [77]. Li *et al.* propose learning a task-driven graph for each graph [101]. Chen *et al.* propose to search a hidden graph structure that augments the initial graph structure [24]. Ding *et al.* present a globally consistent GCN for hyperspectral image classification [30]. Peng *et al.* propose a reverse graph learning to improve the quality of the graph learning for conducting supervised learning and semi-supervised learning [127].

Chapter 3

Robust SVM with adaptive graph learning

3.1 Introduction

Support Vector Machine (SVM) is one of the classical classifiers since it constructs the classifiers by finding the trade-off between the model complexity and the learning capability with limited samples [64, 139]. In the past decades, a large number of SVM methods have been proposed to solve the practical problems. For example, Zhang *et al.* propose a cost-sensitive SVM classifier to achieve the minimal misclassification cost in the multi-classification problem of face recognition [201]. Mygdalis *et al.* propose a graph SVM to first obtain the graphs and then to embed them into the SVM framework [118]. Tang *et al.* integrate feature selection with the SVM framework to simultaneously conduct feature selection and support vector selection [149]. However, there are still limitations in previous SVM methods to be addressed.

First, it is essential to consider the sample importance because different samples show different contributions to the SVM models. Cost-sensitive SVM in [201] considers the sample importance by focusing on it in the whole class, aiming at solving the issue of class imbalance rather than the importance of individual samples. To address the issue of the sample importance, self-paced learning is designed to automatically learn the sample importance by assigning the important samples with the large weights and the unimportant samples with small weights. For example, Zheng *et al.* apply self-paced learning for conducting unsupervised feature selection [209], while Zhu *et al.* investigate the theory of half-quadratic optimization to consider the importance of the individual sample for the clustering task [217].

Second, the graph matrix measuring the correlations between two samples guarantees the data structure to be preserved in the process of model construction. The graph SVM in [118] considers the correlations between data points by ignoring the case where the graph is learned from the original data which often contains noise and outliers. As a result, the learned graph is incorrect so that affecting the quality of the SVM classifier [219]. In the literature, dynamic graph learning [221, 222] has been designed to learn the graph based on the data distribution. For example, [98] and [210] design to learn the dynamic graph from the low-dimensional space of high-dimensional

data to preserve the data structure for conducting spectral feature selection.

Third, the redundant features has been widely demonstrated to influence the model performance in real applications [149]. Hence, it is necessary to reduce the influence of redundant features in the SVM framework. To address this issue, ℓ_1-SVM [159, 216] considers to replace the ℓ_2-norm regularization in traditional SVM with an ℓ_1-norm regularization [16, 141], by assigning the important features with large weights and the unimportant features with very small or even zero weights so that the redundant features are not involved in the process of the model.

In this chapter, we propose a novel robust SVM classifier to simultaneously take into account the above issues. Specifically, based on the traditional SVM framework in [141], we replace the ℓ_2-norm regularization with an $\ell_{2,1}$-norm regularization to conduct both binary classification and multi-class classification on the low-dimensional space of the original data. The low-dimensional space is obtained by iteratively learning the low-dimensional space and the graph. Moreover, we employ self-spaced learning to consider the sample importance for the processes of the classifier construction, the transformation matrix learning for the low-dimensional space and the dynamic graph learning. Furthermore, we propose a primal solution to optimize the proposed SVM framework.

3.2 Related work

In this section, we review the basic SVM method and its variants, and both the graph learning and self-paced learning are introduced in the following.

3.2.1 SVM review

SVM was proposed by Vapnik [158] and was applied for statistical learning theory by Boser *et al.* [10]. The standard SVM focuses on binary classification, but it has involved in multi-class classification [8]. Its purpose is to find the optimal support vector with the maximum distance and to separate every sample from its nearest training samples within the same class.

Actually, SVM and its variants have been proposed for diverse considerations, and they can be divided into three groups, *i.e.,* class-based methods, sample-based methods and feature-based methods. Cost-sensitive SVM is the representative class-based methods which minimizes a symmetric hinge loss function with a cost-sensitive perspective, so it can obtain the important samples for each class and solve the class imbalance problem. For example, Cao *et al.* construct a SVM wrapper method integrating cost-sensitive SVM with evaluation measurement to improve classification results, optimizing misclassification cost and model parameters simultaneously [14]. Gu *et al.* present to combine the chunk incremental learning with cost-sensitive SVM based on hinge loss representation to update the model for each sample and multiple samples simultaneously [52]. Arya *et al.* extend the standard hinge loss function to consider class costs and imbalance traits in SVM, and integrate the risk minimization with probability elicitation on the designed cost-sensitive loss. [72]. Masnadi *et al.* conduct a SVM combining with mathematical-

optimization-based feature selection by considering accumulating asymmetric misclassification costs [9]. Gan *et al.* present a robust cost-sensitive SVM to remove noisy samples and redundant features together [41]. Fu *et al.* propose a cost-sensitive SVM method by embedding the asymmetric linear–exponential loss function into the υ-SVM for class imbalance issue [39].

Sample-based methods focus on the correlations between the samples. For example, graph SVM utilizes the graph structure to express geometric correlations and exploits the intrinsic representations of data during the optimization process. Singh *et al.* use the graph formulation of activities among video data to stand for the information of motion and appearance, and then employ a graph kernel to conduct activity classification so as to detect abnormal actions [147]. Mygdalis *et al.* consider multiplex data correlations based on the SVM framework via combining multiple graph structures with the multi-kernel method [119]. Xie *et al.* take the intrinsic geometry correlations among data into account and employ structured output SVM to rake correlations in the early expression detection model with a graph regularization [173]. Hu *et al.* propose an improved Laplacian SVM method with semi-supervised learning, where hinge loss is consisted of unlabelled data and labelled data and Laplacian matrix is conducted to consider the correlations between the samples [66].

Sparse SVM is an important feature-based method. It extracts the representative feature subset to replace the original data set and conducts the task of classification based on the SVM framework. For example, Xu *et al.* present a scaling factor with a flexible parameter to adaptively adjust the weights of feature distribution during the step of feature selection, and used the subspace constraint to obtain the sparsity of weighting matrix [174]. Shao *et al.* employ sparse ℓ_q-norm regularization combined with least square loss into SVM framework for the classification task [140]. Liu *et al.* propose the ℓ_0-SVM to replace the standard ℓ_2-SVM for feature selection and classification. This way is devised to solve primary and dual variables as well as apply for biological field [108]. Zhou *et al.* employ the sparse ℓ_0-norm to the kernel SVM and obtain the important support vectors resulting in the better performance of data reduction [214]. Tanveer *et al.* propose a twin multi-class classification SVM which use k-nearest neighbor-based way to obtain the sparse samples for different categories [151].

3.2.2 Self-paced learning

Self-paced learning (SPL) is designed by Kumar [94] to replace the heuristic strategy. SPL method defines the significance of the samples with regard to the reconstruction error, *i.e.,* the value of loss function. If the reconstruction error of one sample is less than a setting parameter, the sample is important, otherwise it is unimportant. SPL usually selects a part of samples to initialize the training model, and then it gradually adds more samples to enhance the generalization ability of the training model until the established model achieves stability.

Many SPL methods have been designed to improve the model performance by considering the sample importance or the sample diversity. For example, Gan *et al.* employ the self-paced learning on the hard weight for conducting supervised feature selection [42]. There are two types of representations in SPL based on regularization categories, *i.e.,* explicit way [192, 209] and

implicit way [36, 53]. The explicit way usually updates the regularizer by considering both regularizer and its derivation. For example, Zheng *et al.* conduct an unsupervised feature selection method combined with self-paced learning to select the important samples and features simultaneously [209]. Huang *et al.* utilize self-paced learning to consider the important features and important samples in multi-view clustering method [71]. Wang *et al.* conduct a complementary role of self-paced learning and self-consistency regularization in semi-supervised segmentation [162]. Hence, both of them are necessary for the calculation. In contrast, the implicit way only focuses on the selected regularizer and its related minimization function so that this way provides much compact constraints on the self-paced regularizer. As a result, the minimization function is the essential term. For example, Fan *et al.* conduct a self-paced learning with implicit regularization to obtain the better weighting scheme and the robust performance based on convex conjugacy theory [36]. Zheng *et al.* propose a robust feature selection method which contains the loss function by using self-paced implicit way and the sparse learning way to search the important samples subset and features subset on incomplete data [211]. Yuan *et al.* employ self-paced implicit regularization to adaptively learn the correlations of samples and the signification features [181].

3.3 Method

3.3.1 SVM methods

For n sample-label pairs $(\mathbf{x}^i, y_i)$, $\mathbf{x}^i \in \mathbb{R}^{1\times d}$ and $y_i \in \{-1, +1\}$, and the conventional ℓ_2-SVM is described as

$$\min_{\mathbf{w},b} \ \tfrac{1}{2}||\mathbf{w}||_2^2 + \lambda_1 \sum_i^n (1 - (\mathbf{x}^i\mathbf{w} + b)y_i)_+, \tag{3.1}$$

where $\mathbf{w} \in \mathbb{R}^{d\times 1}$ is the coefficient vector, $b \in \mathbb{R}$ indicates the bias term, $\lambda_1 > 0$ is a penalty parameter and it can adjust the importance between the first term and the second term. The aim of the second term is to measure the difference between two scalars, *i.e.,* $\mathbf{x}^i\mathbf{w} + b$ and y_i, and $(1 - (\mathbf{x}^i\mathbf{w} + b)y_i)_+ = max(1 - (\mathbf{x}^i\mathbf{w} + b)y_i, 0)$.

In order to remove the noise features, ℓ_1-SVM uses an ℓ_1-norm to replace the ℓ_2-norm on the regularization term by

$$\min_{\mathbf{w},b} \ \tfrac{1}{2}||\mathbf{w}||_1 + \lambda_1 \sum_i^n (1 - (\mathbf{x}^i\mathbf{w} + b)y_i)_+. \tag{3.2}$$

3.3.2 Local structure preservation

Although ℓ_1-SVM can set the weight of useless features to zero, the correlations between samples cannot be ignored. Specifically, if two samples have a strong connection, it is expected to be assigned high weight. Hence, we consider to preserve the local importance of the samples. To do this, Laplacian SVM (LapSVM) which conducts the graph Laplacian matrix to represent the

correlations between any two samples is the classic method [7]. However, LapSVM belongs to the fixed graph learning since it constructs the graph matrix from the original data and the learned graph remains the same throughout the entire process. Besides, the graph structure is usually changeable due to the updated transformation matrix during the optimization step. Thus, we employ the dynamic graph learning to replace the fixed graph learning to achieve the optimal local importance. The details are listed as follows:

$$\min_{\mathbf{S}} \sum_{i,j}^{n} ||\mathbf{x}^i - \mathbf{x}^j||_2^2 s_{i,j} + \gamma \sum_{i}^{n} ||\mathbf{s}_i||_2^2, \tag{3.3}$$

where $s_{i,j}$ is the i-th row and j-th column of the similarity matrix $\mathbf{S} \in \mathbb{R}^{n \times n}$ which indicates the correlations between the samples.

The first term in Eq. (3.3) denotes the loss function for generating the similarity matrix. The second term is devised to avoid the trivial solution for the whole equation. However, Eq. (3.3) has to face the issue that noises and outliers are inevitable in the original data, so we transform Eq. (3.3) into the low-dimensional subspace to have

$$\begin{aligned} \min_{\mathbf{S},\mathbf{W}} \ & \sum_{i,j}^{n} ||\mathbf{x}^i\mathbf{W} - \mathbf{x}^j\mathbf{W}||_2^2 s_{i,j} + \gamma \sum_{i}^{n} ||\mathbf{s}_i||_2^2 \\ & s.t., \ \forall i, \ \mathbf{s}_i^T\mathbf{1} = 1, \ s_{i,i} = 0, \ s_{i,j} \geq 0, \ \ if \ \ j \in \mathcal{N}(i), \ \ otherwise \ \ 0, \end{aligned} \tag{3.4}$$

where $\mathbf{W} \in \mathbb{R}^{d \times c}$ denotes the transformation matrix. If the sample $\mathbf{x}^i$ has similar relation to the sample $\mathbf{x}^j$, it is reasonable that their corresponding predictions (*i.e.,* $\mathbf{x}^i = \mathbf{x}^i\mathbf{W}$ and $\mathbf{x}^j = \mathbf{x}^j\mathbf{W}$) also have the same relation. $\mathbf{1}$ and $\mathcal{N}(i)$, respectively, indicate an all-one-element vector and the set of nearest neighbors of the i-th sample. The shift invariant similarity is obtained by the constraint $\mathbf{s}_i^T\mathbf{1} = 1$.

Compared with the traditional fixed graph methods, k is the only parameter in Eq. (3.4) since the similarity matrix is learned from the data, *i.e.,* the more distant samples are, the smaller their values have, and vice verse. Moreover, the graph matrix is learned from 'clean' data by $\mathbf{XW}$ in the low-dimensional space.

3.3.3 Sample importance

Sample importance selects the samples with large weights and the unimportant samples with small values or even zero. In this chapter, we employ Self-paced Learning (SPL) to consider the global correlations among samples, by distributing different weights to different samples. Specifically, this way first selects a part of the samples to build the initial classifier, then gradually adds partial samples to complete the model.

Generally, SPL methods include two different ways, *i.e.,* explicit methods and implicit methods. For simplify, in this chapter, we employ the explicit regularizer to mark the important samples and apply for the ℓ_1-SVM to have the following,

$$\min_{\mathbf{w},b} \ \tfrac{1}{2}||\mathbf{w}||_1 + \lambda_1 \sum_{i}^{n} v^i(1 - (\mathbf{x}^i\mathbf{w} + b)y_i)_+ - \lambda_2||\mathbf{v}||_1, \tag{3.5}$$

where $\mathbf{v} \in \mathbb{R}^{n\times 1}$ indicates a vector for diverse samples. λ_2 plays an important role in the process of learning new samples. Specifically, if the value of λ_2 is small, only samples with small losses are considered for the training process. While λ_2 grows, more samples with larger losses are gradually appended to the training process so that Eq. (3.5) results in robust models.

3.3.4 Objective function

Although the graph matrix $\mathbf{S}$ learnt from the low-dimensional space is constructed, both the graph matrix $\mathbf{S}$ and the matrix $\mathbf{W}$ are not know in advance. Therefore, we combine Eq. (3.4) with Eq. (3.5) into a unified framework. Besides, we apply the united model for the multi-class classification analysis, and use the squared hinge loss function to replace the standard hinge loss function. The details are showed as follows,

$$\begin{aligned}\min_{\mathbf{W},\mathbf{b},\mathbf{v},\mathbf{S}} \ & \tfrac{1}{2}||\mathbf{W}||_{2,1} + \lambda_1 \sum_{i}^{n}\sum_{j}^{c} v^i(1-(\mathbf{x}^i\mathbf{w}_j+b_j)y_{i,j})_+^2 \\ & - \lambda_2||\mathbf{v}||_1 + \lambda_3 \sum_{i,j}^{n} ||\mathbf{x}^i\mathbf{W}-\mathbf{x}^j\mathbf{W}||_2^2 s_{i,j} + \gamma \sum_{i}^{n} ||\mathbf{s}_i||_2^2 \\ s.t., \ & 0 \le v^i \le 1, \ \forall i, \ \mathbf{s}_i^T\mathbf{1} = 1, \ s_{i,i} = 0, \\ & s_{i,j} \ge 0, \ if \ j \in \mathcal{N}(i), \ otherwise \ 0,\end{aligned} \tag{3.6}$$

where λ_1, λ_2, λ_3, and γ are tuning parameters, respectively, to adjust the magnitude of the squared hinge loss function, the learning pace, the adaptive structure, and the similarity between the samples. Specifically, this proposed method can 1) select the robust samples by the second term and the third term; 2) obtain the graph matrix by the fourth term and the fifth term; 3) tune the sparsity of the coefficient matrix by the first term.

Eq. (3.6) iteratively updates the transformation matrix $\mathbf{W}$, the bias term $\mathbf{b}$, the graph matrix $\mathbf{S}$ and the vector $\mathbf{v}$. As a result, given the optimal $\mathbf{W}$, we compute the ℓ_2-norm of $\mathbf{w}^i, i \in [1, 2, ..., d]$, and then sort them in the descending order. Finally, we select top r features according to the top r ranked ℓ_2-norm values as the ultimate result of the proposed spectral feature selection method.

3.3.5 Optimization

Eq. (3.6) is not jointly convex to all variables (*i.e.,* $\mathbf{W}$, $\mathbf{b}$, $\mathbf{v}$, and $\mathbf{S}$), but is convex for every variable while fixing the others. In this chapter, we employ two optimization strategies *i.e.,* Augmented Lagrangian Method (ALM) [46] and Iteratively Re-weighted Least Square (IRLS) [27] to deal with the problems in Section 3.3.5.1 - 3.3.5.3, respectively. We list the pseudo code in Algorithm 3.1.

3.3.5.1 Update $\mathbf{W}$ and $\mathbf{b}$ by fixing $\mathbf{v}$, $\mathbf{S}$

When $\mathbf{v}$ and $\mathbf{S}$ are fixed, the third term and the fifth term can be ignored in Eq. (3.4). Besides, with the observation, *i.e.,* $1-(\mathbf{x}^i\mathbf{w}_j+b_j) = y_{i,j}(y_{i,j}-(\mathbf{x}^i\mathbf{w}_j+b_j))$, we employ the auxiliary

Algorithm 3.1: The pseudo code of solving Eq. (3.6).

Input: $\mathbf{X} \in \mathbb{R}^{n\times d}$, $\mathbf{Y} \in \mathbb{R}^{n\times c}$, k, λ_1, and λ_3;
Output: $\mathbf{W} \in \mathbb{R}^{d\times c}$;
1. Calculate k nearest neighbors of all samples;
2. Initialize $\mathbf{W} \in \mathbf{1}^{d\times c}, \mathbf{b} \in \mathbf{1}^{d\times c}, \mathbf{v} \in \mathbf{1}^{n\times 1}$;
3. Initialize $\mathbf{S}$ by Eq. (3.4);
4. **Repeat:**
   4.1 Update $\mathbf{W}, \mathbf{b}$ via Algorithm 3.2;
   4.2 Update $\mathbf{S}$ via Eq. (3.21);
   4.3 Calculate $\mathbf{L}_s = \mathbf{D}_s - \frac{\mathbf{S}^T+\mathbf{S}}{2}$;
   4.4 Update $\mathbf{v}$ via Eq. (3.26);
until converge

Algorithm 3.2: The pseudo code of solving $\mathbf{W}$ and $\mathbf{b}$.

Input: $\mathbf{X} \in \mathbb{R}^{n\times d}$, $\mathbf{Y} \in \mathbb{R}^{n\times c}$, and θ;
Output: $\mathbf{W} \in \mathbb{R}^{d\times c}$ and $\mathbf{b} \in \mathbb{R}^{1\times c}$;
1. Initialize $\mathbf{W} \in \mathbf{1}^{d\times c}$ and $\mathbf{b} \in \mathbf{1}^{d\times c}$;
2. Initialize $\mu = 0$;
3. **Repeat:**
   3.1 Update $\mathbf{E}$ via Eq. (3.12);
   3.2 Update $\mathbf{W}, \mathbf{b}$ via Eq. (3.15);
   or Choose the fast method: Update $\mathbf{W}, \mathbf{b}$ via Eq. (3.16), and q is computed with Eq. (3.18);
   3.3 Update μ via Eq. (3.10);
until converge

variables $e_{i,j} = y_{i,j} - (\mathbf{x}^i\mathbf{w}_j + b_j), 1 \le i \le n, 1 \le j \le c$ to have:

$$\min_{\mathbf{W},\mathbf{b},\mathbf{E}} \ \tfrac{1}{2}||\mathbf{W}||_{2,1} + \lambda_1 \sum_i^n \sum_j^c v^i (y_{i,j} e_{i,j})_+^2 + \lambda_3 \sum_{i,j}^n ||\mathbf{x}^i\mathbf{W} - \mathbf{x}^j\mathbf{W}||_2^2 s_{i,j}. \tag{3.7}$$

Based on the ALM method, we construct the Lagrangian function of Eq. (3.7) to have:

$$\begin{aligned}\mathcal{L}(\mathbf{W}, \mathbf{b}, \mathbf{E}, \mu) = \ & \lambda_1 \sum_i^n \sum_j^c v^i (y_{i,j} e_{i,j})_+^2 + v^i \mu^T ||\mathbf{X}\mathbf{W} + \mathbf{1}_n\mathbf{b} - \mathbf{Y} + \mathbf{E}||_F^2 \\ & + \tfrac{1}{2}||\mathbf{W}||_{2,1} + \lambda_3 \sum_{i,j}^n ||\mathbf{x}^i\mathbf{W} - \mathbf{x}^j\mathbf{W}||_2^2 s_{i,j},\end{aligned} \tag{3.8}$$

where $\mathbf{X} \in \mathbb{R}^{n\times d} = [x^1, x^2, ..., x^n]$, $\mathbf{1}_n \in \mathbb{R}^{n\times 1} = [1, 1, ..., 1]^T$ is an all-one-element vector, $\mathbf{Y} \in \mathbb{R}^{n\times c} = [y_1, y_2, ..., y_c]$, $\mathbf{E} \in \mathbb{R}^{n\times c} = [e_1, e_2, ..., e_c]$. The third term is the point-wise multiplication of the amount of violation of the n constraints $(\mathbf{x}^i\mathbf{w}_j + b_j) - y_{i,j} + e_{i,j} = 0$ with the vector $\mu \in \mathbb{R}^n$ consisting of n Lagrangian multipliers.

By considering the ALM method, the augmented Lagrangian function of Eq. (3.7) is:

$$\begin{aligned}\mathcal{A}_{\mathcal{L}}(\mathbf{W},\mathbf{b},\mathbf{E},\mu,\theta) = \ & \lambda_1 \sum_{i}^{n}\sum_{j}^{c} v^i (y_{i,j}e_{i,j})_+^2 + v^i\tfrac{\theta}{2}||\mathbf{XW}+\mathbf{1}_n\mathbf{b}-\mathbf{Y}+\mathbf{E}+\tfrac{\mu}{\theta}\mathbf{1}||_F^2 \\ & + \tfrac{1}{2}||\mathbf{W}||_{2,1} + \lambda_3 \sum_{i,j}^{n} ||\mathbf{x}^i\mathbf{W}-\mathbf{x}^j\mathbf{W}||_2^2 s_{i,j},\end{aligned} \tag{3.9}$$

where $\mathbf{1}$ denotes a matrix and all elements are equivalent to 1, and Eq. (3.9) is close to zero when $\theta \to \infty$, and eventually it becomes negligible.

At the t-th iteration, the Lagrangian multiplier vector μ can be updated by:

$$\mu_{(t)} = \mu_{(t-1)} + \theta_{(t)}||\mathbf{XW}+\mathbf{1}_n\mathbf{b}-\mathbf{Y}+\mathbf{E}||_F^2. \tag{3.10}$$

The parameter θ which indicates the augmented penalty is monotonically non-decreasing over the iterative process. Section 3.3.8.1 discusses how to determine the relating series $\theta_{(t)}$ in each iteration t.

At every iteration, we can split the updating of $\mathbf{W}$, $\mathbf{b}$, $\mathbf{E}$ into two independent parts: 1) to update $\mathbf{E}$ by fixing $\mathbf{W}$ and $\mathbf{b}$, and 2) to update $\mathbf{W}$ and $\mathbf{b}$ by fixing $\mathbf{E}$.

- 1) Update $\mathbf{E}$ by fixing $\mathbf{W}$ and $\mathbf{b}$

  When $\mathbf{W}$ and $\mathbf{b}$ are fixed, each $e_{i,j}$ can be computed by:

$$\begin{aligned} e_{i,j} &= \arg\min_{e_{i,j}} \ G_{i,j}(e_{i,j}) \\ &= \arg\min_{e_{i,j}} \ \lambda_1 \sum_{i}^{n}\sum_{j}^{c} v^i (y_{i,j}e_{i,j})_+^2 \\ &\quad + v^i\tfrac{\theta}{2}||e_{i,j}-(y_{i,j}-\mathbf{x}^i\mathbf{w}_j-b_j-\tfrac{\mu_i}{\theta})||^2 \\ &= \arg\min_{e_{i,j}} \ \alpha \sum_{i}^{n}\sum_{j}^{c} v^i (y_{i,j}e_{i,j})_+^2 + \tfrac{1}{2}(e_{i,j}-t_{i,j})^2, \end{aligned} \tag{3.11}$$

  where $\alpha = \frac{\lambda_1}{\theta}$ and $t_{i,j} = y_{i,j} - \mathbf{x}^i\mathbf{w}_j - b_j - \frac{\mu_i}{\theta}$. Eq. (3.11) is easy to be solved because it only needs to find out the minimum value in both $e_{i,j} \leq 0$ and $e_{i,j} > 0$ when $e_{i,j}$ acts as the minimizer for the function $G_{i,j}(e_{i,j})$. Similarly, when the condition is changed into $y_{i,j}e_{i,j}$, the result is showed as follows:

  - When $y_{i,j}e_{i,j} \leq 0$, the result is $(y_{i,j}e_{i,j})_+^2 = 0$, and the smaller value is selected between $G_{i,j}(0)$ and $G_{i,j}(t_{i,j})$.
  - When $y_{i,j}e_{i,j} > 0$, the result is obtained by calculating the partial derivative of $e_{i,j}$ and let its equation to zero: $\frac{\partial G_{i,j}}{\partial e_{i,j}} = 2\alpha v^i(y_{i,j}e_{i,j}) + e_{i,j} - t_{i,j} = 0$

  It is difficult to solve the above condiction because of the arbitrary given α. However, with the constraint $\alpha > 0$, $\frac{\partial G_{i,j}}{\partial e_{i,j}}$ is monotonically increasing and we can use the well-known binary search method to narrow the possible range of $e_{i,j}$ by half at every operation, and

obtain an ε-accurate solution in $O(\log \frac{1}{\varepsilon})$ time. Therefore, the explicit solution can be derived as:

$$\begin{cases} e_{i,j} = \frac{t_{i,j}}{1+2\alpha v^i}, & y_{i,j}t_{i,j} > 0, \\ e_{i,j} = t_{i,j}, & y_{i,j}t_{i,j} \leq 0. \end{cases} \tag{3.12}$$

- 2) Update $\mathbf{W}$ and $\mathbf{b}$ by fixing $\mathbf{E}$

  When $\mathbf{E}$ is fixed, we obtain the objective function with respect to $\mathbf{W}$ and $\mathbf{b}$ as follows:

$$\min_{\mathbf{W},\mathbf{b}} \theta^{-1}||\mathbf{W}||_{2,1} + v^i||\mathbf{X}\mathbf{W} + \mathbf{1}_n\mathbf{b} - \mathbf{Z}||_F^2 + \lambda_3 \sum_{i,j}^{n} ||\mathbf{x}^i\mathbf{W} - \mathbf{x}^j\mathbf{W}||_2^2 s_{i,j}, \tag{3.13}$$

where $\mathbf{Z} = \mathbf{Y} - \mathbf{E} - \frac{\mu}{\theta}\mathbf{1}$. Besides, different classes are independent to each other, we can change Eq. (3.13) to:

$$\min_{\mathbf{w}_j,b_j} \theta^{-1}\mathbf{w}_j^T\mathbf{R}\mathbf{w}_j + v^i||\mathbf{X}\mathbf{w}_j + \mathbf{1}_n b_j - \mathbf{z}_j||_2^2 + \lambda_3\mathbf{w}_j^T\mathbf{X}^T\mathbf{L}_s\mathbf{X}\mathbf{w}_j, \tag{3.14}$$

where a diagonal matrix $\mathbf{R}$ and its diagonal elements can be defined as $r_{i,i} = \frac{1}{2||\mathbf{w}_j||_2^2}, 1 \leq i \leq d$, $\mathbf{L}_s = \mathbf{D}_s - \frac{\mathbf{S}^T+\mathbf{S}}{2}$ is the Laplacian matrix of graph structure. As a result, Eq. (3.14) can be converted into a standard Least Square Regression (LSR) problem as follows, by setting $\mathbf{A} = [\mathbf{X}^T, \mathbf{1}_n; \theta^{-\frac{1}{2}}\mathbf{R} + \lambda_3{}^{\frac{1}{2}}\mathbf{X}^T\mathbf{L}_s\mathbf{X}, \mathbf{0}]$, $\mathbf{m} = [\mathbf{w}_j; b_j]$, $\mathbf{u} = [\mathbf{z}_j; \mathbf{0}]$, we have

$$\min \ ||\mathbf{A}\mathbf{m} - \mathbf{u}||^2. \tag{3.15}$$

Eq. (3.15) can be solved by the default LSQR function in MATLAB [123], and its time complexity is $O(n\hat{d}^2)$ where $\hat{d}$ indicates the average number of non-zero elements of each sample. Therefore, we derive the optimal solution of $\mathbf{w}_j$ and b_j as follows,

$$\begin{cases} \mathbf{w}_{j_h} = \mathbf{X}^T\mathbf{V}(\mathbf{X}\mathbf{w}_j + \mathbf{1}_n b_j - \mathbf{z}) + \theta^{-1}\mathbf{R}\mathbf{w}_j + \lambda_3\mathbf{X}^T\mathbf{L}_s\mathbf{X}\mathbf{w}_j, \\ b_{j_h} = \mathbf{1}_n^T\mathbf{V}\mathbf{1}_n b_j + \mathbf{1}_n^T\mathbf{V}(\mathbf{X}\mathbf{w}_j - \mathbf{z}). \end{cases} \tag{3.16}$$

The optimal step-size q can be obtained by the following quadratic function,

$$\begin{aligned} \min_q \ & \theta^{-1}(\mathbf{w}_j - q\mathbf{w}_{j_h})^T\mathbf{R}(\mathbf{w}_j - q\mathbf{w}_{j_h}) \\ & + v^i||\mathbf{X}(\mathbf{w}_j - q\mathbf{w}_{j_h}) + \mathbf{1}_n(b_j - qb_{j_h}) - \mathbf{z}||_2^2 \\ & + \lambda_3(\mathbf{w}_j - q\mathbf{w}_{j_h})^T\mathbf{X}^T\mathbf{L}_s\mathbf{X}(\mathbf{w}_j - q\mathbf{w}_{j_h}). \end{aligned} \tag{3.17}$$

And it has the explicit solution:

$$\begin{aligned} q = (\mathbf{w}_{jh}^T\mathbf{w}_{jh} + b_{jh}^T b_{jh})((\mathbf{X}\mathbf{w}_{j_h} + \mathbf{1}_n b_{j_h})^T\mathbf{V}(\mathbf{X}\mathbf{w}_{jh} + \mathbf{1}_n b_{jh}) \\ + \theta^{-1}\mathbf{w}_{jh}^T\mathbf{R}\mathbf{w}_{j_h} + \lambda_3\mathbf{w}_{jh}^T\mathbf{X}^T\mathbf{L}_s\mathbf{X}\mathbf{w}_{j_h})^{-1}. \end{aligned} \tag{3.18}$$

The goal of above equality is to simplify the calculation of q. In this way, we can yield $\mathbf{W}, \mathbf{b}$ with different methods and the detail of optimizing $\mathbf{W}$ and $\mathbf{b}$ is listed in Algorithm 3.2.

3.3.5.2 Update S by fixing v, W and b

If we want to solve $\mathbf{S}$, Eq. (3.6) becomes:

$$\begin{aligned} \min_{\mathbf{S}} \ & \lambda_3 \sum_{i,j}^{n} ||\mathbf{x}^i\mathbf{W} - \mathbf{x}^j\mathbf{W}||_2^2 s_{i,j} + \gamma \sum_{i}^{n} ||\mathbf{s}_i||_2^2 \\ & s.t., \ \forall i, \ \mathbf{s}_i^T\mathbf{1} = 1, \ s_{i,i} = 0, \ s_{i,j} \geq 0, \ \ if \ \ j \in \mathcal{N}(i), \ \ otherwise \ \ 0, \end{aligned} \tag{3.19}$$

where $\mathbf{s}_i \in \mathbb{R}^{n\times 1}$ indicates a vector with the j-th element as $s_{i,j}$, γ is the tuning parameter. The optimal solution is that all data points can be the neighbors of $\mathbf{x}^i\mathbf{W}$ with the same probability $\frac{1}{n}$, which can be viewed as a prior in the neighbors assignment.

We need to obtain k nearest neighbors of all samples via computing their Euclidean distance, and setting the value of $s_{i,j}$ equal to 0 when the j-th sample does not belong to one of k-nearest neighbors of the i-th sample. Besides, the optimization of $\mathbf{S}$ is equal to optimize every vector $\mathbf{s}_i, i \in [1, n]$ independently, we further convert Eq. (3.19) to individual parts as follows:

$$\min_{\mathbf{s}_i^T\mathbf{1}=1, s_{i,i}=0, s_{i,j}\geq 0} \ \sum_{i,j}^{n} (\lambda_3 ||\mathbf{x}^i\mathbf{W} - \mathbf{x}^j\mathbf{W}||_2^2 s_{i,j} + \gamma s_{i,j}^2). \tag{3.20}$$

Let $d_{i,j} = \lambda_3 ||\mathbf{x}^i\mathbf{W} - \mathbf{x}^j\mathbf{W}||_2^2$, and $\mathbf{d}_i \in \mathbb{R}^{n\times 1}$ as a vector with the j-th element as $d_{i,j}$, Eq. (3.20) can be written in a vector form as follows:

$$\min_{\mathbf{s}_i^T\mathbf{1}=1, s_{i,i}=0, s_{i,j}\geq 0} \ ||\mathbf{s}_i + \tfrac{1}{2\gamma}\mathbf{d}_i||_2^2. \tag{3.21}$$

Actually, the regularization parameter is usually difficult to tune, in this chapter, we propose an effective approach to deal with the γ in Eq. (3.21). Specifically, for every i, we conduct the corresponding Lagrangian function as follows:

$$||\mathbf{s}_i + \frac{1}{2\gamma}\mathbf{d}_i||_2^2 - \sigma(\mathbf{s}_i^T\mathbf{1} - 1) - \boldsymbol{\tau}^T\mathbf{s}_i, \tag{3.22}$$

where $\sigma \in \mathbb{R}$ represents a Lagrange multiplier and $\boldsymbol{\tau} \in \mathbb{R}_+^n$ denotes non-negative vector Lagrange multipliers.

According to the complementary slackness Karush-Kuhn-Tucker (KKT) conditions [11], for any $\mathbf{s}_i > 0$, we have $\tau_i = 0$ and the followings

$$s_{i,j} = (-\frac{d_{i,j}}{2\gamma_i} + \sigma)_+. \tag{3.23}$$

The parameter of γ can be adjusted automatically in Section 3.3.8.2.

3.3.5.3 Update v by fixing S, W, and b

If we want to solve the $\mathbf{v}$, Eq. (3.6) can be converted into:

$$\min_{\mathbf{v}} \ \lambda_1 \sum_{i}^{n} \sum_{j}^{c} v^i((1-(\mathbf{x}^i\mathbf{w}_j+b_j)y_{i,j})_+^2) - \lambda_2||\mathbf{v}||_1 \ \ s.t.\ 0 \leq v^i \leq 1, \tag{3.24}$$

where λ_2 denotes a parameter for controlling the learning pace. Eq. (3.24) illustrates the loss of one instance is discounted by one weight.

Defining $\mathbf{L} = \sum_i^n L_i = \sum_i^n \sum_j^c (1-(\mathbf{x}^i\mathbf{w}_j+b_j)y_{i,j})_+^2$, we have:

$$\min_{\mathbf{v}} \ \sum_{i}^{n}(\lambda_1 v^i L_i - \lambda_2 v^i) \ \ s.t.,\ 0 \leq v^i \leq 1. \tag{3.25}$$

According to Eq. (3.25), the corresponding global optimum v^i can be easily calculated by:

$$v^i = \begin{cases} 1, & L_i \leq \lambda_2, \\ 0, & otherwise. \end{cases} \tag{3.26}$$

There exists an intuitive explanation behind this alternative search strategy. Specifically, if $\mathbf{v}$ is updated, every sample whose loss is smaller than a certain threshold λ_2 is taken as an important sample while training (*i.e.,* $v^i = 1$), otherwise we do not select it (*i.e.,* $v^i = 0$).

3.3.6 Convergence analysis

By denoting the objective function value of Eq. (3.6) as $F(\mathbf{W}, \mathbf{b}, \mathbf{v}, \mathbf{S})$ and the t-th iteration of the variables as $\mathbf{W}^{(t)}$, $\mathbf{b}^{(t)}$, $\mathbf{v}^{(t)}$, and $\mathbf{S}^{(t)}$, we prove the convergence of our proposed method as follows.

Based on the literature [27, 174] and the fixed $\mathbf{v}^{(t)}$ and $\mathbf{S}^{(t)}$ in the process of optimization, we have:

$$F(\mathbf{W}^{(t+1)}, \mathbf{b}^{(t+1)}, \mathbf{v}^{(t)}, \mathbf{S}^{(t)}) \leq F(\mathbf{W}^{(t)}, \mathbf{b}^{(t)}, \mathbf{v}^{(t)}, \mathbf{S}^{(t)}). \tag{3.27}$$

While fixing $\mathbf{W}^{(t+1)}, \mathbf{b}^{(t+1)}$, and $\mathbf{v}^{(t)}$, the optimization with respect to $\mathbf{S}$ has the following inequality:

$$F(\mathbf{W}^{(t+1)}, \mathbf{b}^{(t+1)}, \mathbf{v}^{(t)}, \mathbf{S}^{(t+1)}) \leq F(\mathbf{W}^{(t+1)}, \mathbf{b}^{(t+1)}, \mathbf{v}^{(t)}, \mathbf{S}^{(t)}). \tag{3.28}$$

According to SPL theory [115] and the fixed $\mathbf{W}^{(t+1)}$, $\mathbf{b}^{(t+1)}$, and $\mathbf{S}^{(t+1)}$, we have:

$$F(\mathbf{W}^{(t+1)}, \mathbf{b}^{(t+1)}, \mathbf{v}^{(t+1)}, \mathbf{S}^{(t+1)}) \leq F(\mathbf{W}^{(t+1)}, \mathbf{b}^{(t+1)}, \mathbf{v}^{(t)}, \mathbf{S}^{(t+1)}). \tag{3.29}$$

By combining Eq. (3.27), Eq. (3.28) with Eq. (3.29), the final inequality is:

$$F(\mathbf{W}^{(t+1)}, \mathbf{b}^{(t+1)}, \mathbf{v}^{(t+1)}, \mathbf{S}^{(t+1)}) \leq F(\mathbf{W}^{(t)}, \mathbf{b}^{(t)}, \mathbf{v}^{(t)}, \mathbf{S}^{(t)}). \tag{3.30}$$

We proved that the value of objective function in Eq. (3.6) decreases for every iteration.

3.3.7 Complexity analysis

In every iteration, Algorithm 3.2 only needs three matrix-by-vector multiplications with complexity $O(n\hat{d})$, where $\hat{d}$ shows the average number of non-zero elements per instance. In the classification task, the high-dimensional features are always reduced by the pre-screening procedure, hereby $\hat{d}$ is not large. As a result, our proposed algorithm has linear computational cost with regard to the sample number n.

The time cost of Algorithm 3.1 focuses on the computation cost of $d_{i,j}$ in Eq. (3.23) and $v_{i,i}$ in Eq. (3.26), so their corresponding complexity are $O(nd^2)$ and $O(n)$, where *n* and *d* denote the number of the samples and the features, respectively. In our experiments, our method generally converges within t iterations, *e.g.,* 30, so the total complexity of the proposed method is $O(tnd^2)$.

3.3.8 Parameters' determination

3.3.8.1 The determination of $\theta_{(t)}$

In order to prove the sequence $\mu_{(t)}$ in Algorithm 3.2 is bounded, we introduce Lemma 1 [122] as follows.

Lemma 1. *If the sequences* $\{\mathbf{w}_j^{(t)}\}$, $\{\mathbf{b}_{(t)}\}$ *and* $\{\mathbf{e}_j^{(t)}\}$ *denote the optimal solution, denoting each iteration* $t > 0$ *and* $\sum_t^{\infty} \frac{\theta(t+1)}{\theta(t)^2} < \infty$, *we have*

$$\begin{aligned}&\phi(\mathbf{w}_j^{(t+1)}) + ||\lambda_1 \mathbf{v}(\mathbf{y}_j \mathbf{e}_j^{(t+1)})_+||^2 + \tfrac{\theta_{(t+1)}}{2}||\mathbf{X}\mathbf{w}_j^{(t+1)} + \mathbf{1}\mathbf{b}_{(t)} - \mathbf{y}_j + \mathbf{e}_j^{(t+1)}||^2 + \varphi(\mathbf{w}_j^{(t+1)}) \\ &\leq \phi(\mathbf{w}_j^{(t)}) + ||\lambda_1 \mathbf{v}(\mathbf{y}_j \mathbf{e}_j^{(t)})_+||^2 + \tfrac{\theta_{(t)}}{2}||\mathbf{X}\mathbf{w}_j^{(t)} + \mathbf{1}\mathbf{b}_{(t)} - \mathbf{y}_j + \mathbf{e}_j^{(t)}||^2 + \varphi(\mathbf{w}_j^{(t)}),\end{aligned}$$

where

$$\begin{cases} \phi(\mathbf{w}_j^{(t)}) = \frac{1}{2} tr((\mathbf{w}_j^{(t)})^T \mathbf{R} \mathbf{w}_j^{(t)}), \\ \varphi(\mathbf{w}_j^{(t)}) = \lambda_3 \sum_{i,j}^{n} ||\mathbf{x}^i \mathbf{w}_j^{(t)} - \mathbf{x}^j \mathbf{w}_j^{(t)}||_2^2 s_{i,j}. \end{cases}$$

By considering Eq. (3.11) and Eq. (3.14), we have the following inequality:

$$\begin{aligned}&\mathcal{A}_{\mathcal{L}}(\mathbf{w}_j^{(t)}, \mathbf{b}_{(t)}, \mathbf{e}_j^{(t)}, \mu_{(t-1)}, \theta_{(t)}) \\ &\leq \mathcal{A}_{\mathcal{L}}(\mathbf{w}_j^{(t-1)}, \mathbf{b}_{(t-1)}, \mathbf{e}_j^{(t)}, \mu_{(t-1)}, \theta_{(t)}) \\ &\leq \mathcal{A}_{\mathcal{L}}(\mathbf{w}_j^{(t-1)}, \mathbf{b}_{(t-1)}, \mathbf{e}_j^{(t-1)}, \mu_{(t-1)}, \theta_{(t)}),\end{aligned} \tag{3.31}$$

which can ensure that the sequence $\{\theta_{(t)}\}$ is non-decreasing.

Hence, based on Lemma 1, we can generate the sequence $\{\theta_{(t)}\}$ and the update method as follows:

$$\begin{aligned}\theta_{(t+1)} = (&\tfrac{\theta_{(t)}}{2}||\mathbf{X}\mathbf{w}_j^{(t)} + \mathbf{1}\mathbf{b}_{(t)} - \mathbf{y}_j + \mathbf{e}_j^{(t)}||^2 + \phi(\mathbf{w}_j^{(t)}) + ||\lambda_1 \mathbf{v}(\mathbf{y}_j \mathbf{e}_j^{(t)})_+||^2 \\ &+ \varphi(\mathbf{w}_j^{(t)}) - \phi(\mathbf{w}_j^{(t+1)}) - ||\lambda_1 \mathbf{v}(\mathbf{y}_j \mathbf{e}_j^{(t+1)})_+||^2 \\ &- \varphi(\mathbf{w}_j^{(t+1)})(\tfrac{1}{2}||\mathbf{X}\mathbf{w}_j^{(t+1)} + \mathbf{1}\mathbf{b}_{(t)} - \mathbf{y}_j + \mathbf{e}_j^{(t+1)}||^2)^{-1}.\end{aligned} \tag{3.32}$$

3.3.8.2 The determination of γ

It is not simple using Eq. (3.23) to find all the suitable $\mathbf{s}_i$. Based on [32], we have Lemma 2 and Lemma 3 as follows.

Lemma 2. *If $\mathbf{s}_i$ achieves the optimal solution in Eq. (3.23) and $\mathbf{s}_i = 0$, letting i and j be two indices and $\boldsymbol{\beta}_i > \boldsymbol{\beta}_j$ ($\boldsymbol{\beta} = -\frac{1}{2\gamma}\mathbf{d}_i$), $\mathbf{s}_j$ equals to zero.*

Based on Lemma 2, there exist some integers $\boldsymbol{I} = [\rho], 1 \leq \rho \leq n$ to meet the non-zero components of the sorted optimal solution, so we have

$$\sigma = \frac{1}{\rho}(\sum_{i}^{\rho} \beta_i - 1), \tag{3.33}$$

As a result, the optimal $\mathbf{s}_i$ can be described as $\mathbf{s}_i = max\{\beta_i - \rho, 0\}$. Besides, Lemma 3 is used to automatically find the optimal ρ.

Lemma 3. *Letting $\boldsymbol{\eta}$ represent the vector after sorting $\boldsymbol{\beta}$ with a descending order, the number of strictly non-negative elements in $\boldsymbol{\alpha}$ is $\rho = max\left\{\eta_j - \frac{1}{j}(\sum_{i=1}^{j=n} \eta_i - 1) > 0\right\}$.*

To consider $\mathbf{s}_i^T\mathbf{1} = 1$ and Eq. (3.33) together, γ can be calculated by $\gamma \leq \frac{\rho}{2}d_{i,\rho+1} - \frac{1}{2}\sum_j^{\rho} d_{i,j}$. Hence, in order to achieve the final value of the γ, we consider to use the mean-value method of n subjects to have

$$\gamma = \frac{1}{n}\sum_{i}^{n}(\frac{2}{\rho}d_{i,\rho+1} - \frac{1}{2}\sum_{j}^{\rho} d_{i,j}), \tag{3.34}$$

where ρ denotes the exact non-zero values and it can be automatically obtained.

3.4 Experiments

We evaluate our proposed method compared with one baseline method and seven state-of-the-art methods on four synthetic data sets and sixteen public real data sets in terms of different classification tasks, *i.e.,* binary classification and multi-class classification tasks.

3.4.1 Data sets

We design four synthetic classification data sets based on the scikit-learn (sklearn) toolbox using the software of python [125], to test our proposed method, *i.e.,* two for binary classification and other two for multiple categories classification. Specifically, we employ the function[1] to generate synthetic data sets, and then use two same data sets for both binary and multi-class classification, *i.e.,* one data set has 1500 samples with 500 dimensions for representation, the other has 2000 samples with 1000 dimensions for representation. We set the term 'redundancy' as 10% and 40%

[1] https://scikit-learn.org/stable/modules/generated/sklearn.datasets.make_classification.html

Table 3.1: The details of synthetic data sets.

Data Sets	Synb1	Synb2	Synm1	Synm2
Samples	1500	2000	1500	2000
Dimensions	500	1000	500	1000
Classes	2	2	3	3
Redundancy	10%	40%	15%	45%
Minimum	-4.84	-6.20	-6.81	-7.86
Maximum	4.92	6.66	7.47	7.92

Table 3.2: The details of real-world data sets.

Data Sets	#(Samples)	#(Dimensions)	#(Classes)	#(Type)
Parkin	1040	26	2	Biology
Spambase	4601	57	2	Email
Hillv	606	100	2	Artificial
German	1000	20	2	Financial
Colon	62	2000	2	Biology
Dbworld	64	4702	2	Text
Madelon	2000	500	2	Artificial
Pcmac	1943	3289	2	Text
Wavef	2746	21	3	Artificial
Lung	203	3312	5	Biology
Ecoli	336	343	8	Biology
Cane	1080	856	9	Text
Warpar	130	2400	10	Image
Yale32	165	1024	15	Image
Coil	1440	1024	20	Image
Isolet	1559	617	26	Text

for binary data sets and 15% and 45% for multi-class classification. The details of synthetic data sets are listed in Table 3.1 and the classification results are reported in Table 3.3.

We download eight binary-class data sets and eight multi-class benchmark data sets from public website and listed their details in Table 3.2. In specific, the data sets (such as Parkin, Spambase, Hillv, German, Wavef, Ecoli, Cane and Isolet) and the data sets (such as Colon, Dbworld, Madelon, Pcmac, Lung, Warpar, Yale32 and Coil), respectively, from UCI Machine Learning Repository[2] and the website of Feature Selection Data sets[3].

These data sets come from six kinds of applications, such as biological data (such as Parkin, Colon, Lung and Ecoli), image data (such as Warpar, Yale32 and Coil), artifical data (scuh Hillv, Madelon and Wavef), text data (such as Dbworld, Pcmac, Cane and Isolet), email data (a mixture data including text and image *e.g.,* Spambase) and financial data (such as German). Moreover, the number of samples is from 62 to 4601 and the number of features is from 20 to 4702. In particular, the number of samples of four data sets is larger than the number of features, such as

[2]https://archive.ics.uci.edu/ml/index.php.
[3]http://featureselection.asu.edu/datasets.php.

Parkin, Spambase, German and Wavef.

3.4.2 Comparison methods

The comparison methods included one Baseline method, three linear SVM methods, and four feature selection methods.

- **Baseline** uses all features to conduct the classification tasks with the SVM classifier. Baseline does not have the process of pre-processing of the original data and can indicate the initial distribution among the original data.
- SVM with the ℓ_1-norm (**L1SVM**) [182] uses the ℓ_1-norm regularization to replace the ℓ_2-norm work on the penalty term (*i.e.,* the squared hinge loss function), and can obtain a sparse performance and avoid the issue of over-fitting.
- General framework for Sparsity Regularization (**GSR**) [126] constructs the $\ell_{2,p}^p$-norm to act on both the least square loss and the regularization term, as well as to flexibly control different sparse degrees on both the loss function term and the regularization term through the related parameter.
- Learning Graph and Self-paced method (**LGS**) [210] integrates diverse techniques, such as dynamic graph construction, feature selection and a subspace constraint in a unified framework, to synchronously consider the correlations between the samples and the correlations between the features.
- Multiclass Kernel-based Vector Machines (**MKVM**) [25] is a classical non-linear SVM method for the multi-class classification by first developing basic framework for multi-class SVM and then considering one slack variable for every sample.
- Cost Sensitive with Laplacian svm (**CSL**) [170] applies the cost sensitivity (*i.e.,* misclassification cost) for the laplacian SVM model to consider the effectiveness between positive data and negative data. In this chapter, we do not take into account the cost-sensitive so that the relevant value is set to one.
- Softmax Regression with Self-paced Learning (**SRSL**) [130] combines the softmax regression with the self-paced learning in a unified framework for multi-class classification.
- Risk Minimization Multiclass method (**RMM**) [99] utilizes the ℓ_p-norm to control the complexity of the related data-dependent bound among the classes under the basic multi-class SVM framework.

The comparison methods include two conventional feature selection methods (such as GSR and LGS), four multi-class SVM methods with different considerations (such as MKVM, CSL, SRSL and RMM). We also regard the method (*i.e.,* Baseline) using all features to conduct classification tasks with the traditional SVM classifier.

Besides, we choose the above comparison methods for two goals, first, as for binary classification, the main purpose is to compare the difference between various loss functions (*i.e.,* the hinge

loss function and the least square loss function), and the dissimilarity among the norm variety (*i.e.,* ℓ_1, $\ell_{2,1}$, and $\ell_{2,p}^p$). Second, in terms of multi-class classification, the key goal is to consider multiple constraints (*i.e.,* self-paced learning, cost-sensitive consideration and risk minimization) to deal with multi-class problems.

3.4.3 Setting

Our experiments work on CPU W-2104 4 cores and 32G RAM within Win10 system, and MATLAB 2019a. We first utilize all methods to obtain the reconstructed weight matrix, and then run the SVM classifier on the representative samples and features to conduct the task of classification.

We employ the 10-fold cross validation method to conduct experiments for all the methods. Specifically, in every experiment, we firstly use every dimensionality reduction method to reduce the dimensions of the training data, and then conduct classification using SVM on the reduced data. In each experiment, we partition the whole data set into ten subsets where 9 subsets are used for training and the left one subset is used for testing. During the training process, we use a 5-fold cross validation method to conduct model selection. In model selection, we set the range of parameters *i.e.,* λ_1 and λ_3 as $(10^{-3}, ..., 10^3)$ for all methods to make fair comparison, where all the methods obtain their best performance. As for the parameter of λ_2, for simplify in this experiment, it can be automatically updated through this way, *i.e.,* $\lambda_2 = V(L)/n$, where $V(L)$ denotes the value of $\mathbf{L}$ and the definition of $\mathbf{L}$ is between Eq. (3.24) and Eq. (3.25). γ can be adjusted automatically in Section 3.3.8.2. Finally, we repeat each experiment ten times and report the final results as the average of all ten times.

We evaluate all methods by using classification accuracy (ACC) for both binary classification and multi-class classification. We also employ other three evaluation metrics (such as sensitivity (SEN), specificity (SPE) and Area Under Curve (AUC)) to evaluate the performance of binary classification, *i.e.,* $ACC = \frac{TP+TN}{TP+TN+FP+FN}$, $SEN = \frac{TP}{TP+FN}$, $SPE = \frac{TN}{TN+FP}$, where TP, TN, FP, FN, respectively, means true positive, true negative, false positive and false negative.

The goal of ACC is to measure the percentage of samples correctly classified among all samples. SEN and SPE, respectively, indicate the correct results of positive sample among the positive samples and the correct results of negative sample among the negative samples. AUC denotes the probability that the classifier assigns a high score to a randomly selected positive sample than to a randomly selected negative sample.

The higher the result of each evaluation metric is, the better the results of the method is, and vice versa.

3.4.4 Result analysis

For the synthetic data sets, our proposed method achieve the best results, followed by CSL, RMM, SRSL, MKVM, LGS, L1SVM, and GSR. This indicates the effectiveness of our proposed method. Specifically, first, multi-class methods (*i.e.,* MKVM, CSL, SRSL and RMM) outperform traditional feature selection methods (*i.e.,* GSR and LGS). Second, the graph structure

Table 3.3: Classification accuracy (ACC) of all methods on four synthetic data sets (%).

Data Sets	Synb1	Synb2	Synm1	Synm2
Baseline	94.41	71.55	67.33	54.42
L1SVM	94.67	74.42	72.49	55.00
GSR	66.00	60.19	58.06	46.22
LGS	92.29	91.55	69.29	47.48
MKVM	89.43	85.11	77.64	76.90
CSL	98.27	93.34	88.27	81.16
SRSL	90.56	90.14	87.92	82.99
RMM	98.40	91.85	85.52	81.57
Proposed	**98.93**	**93.40**	**90.07**	**84.47**

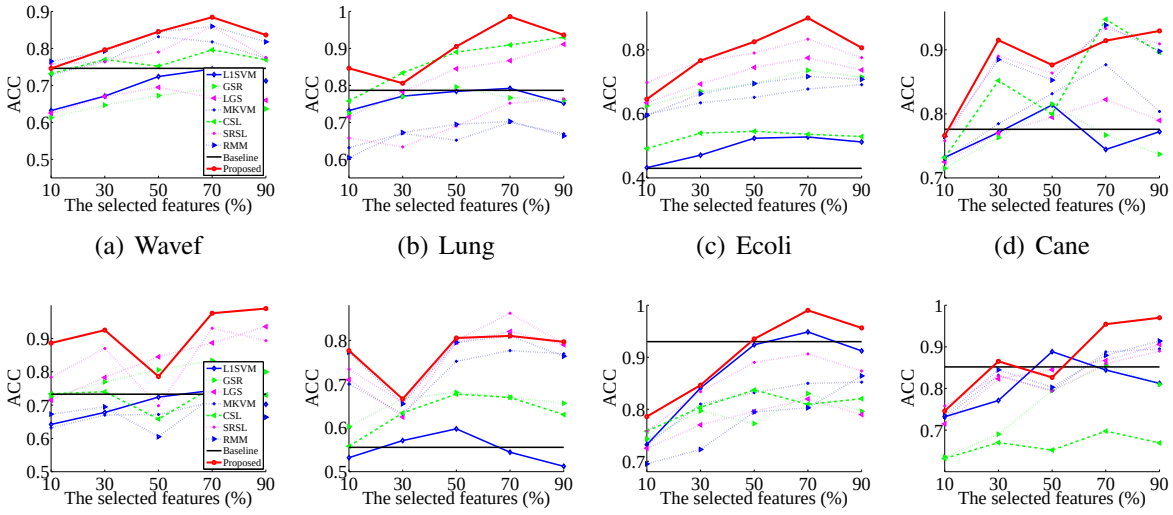
Table 3.4: Classification accuracy (%) of all methods for multi-class classification on eight real data sets.

Data Sets	Wavef	Lung	Ecoli	Cane	Warpar	Yale32	Coil	Isolet
Baseline	74.61	78.67	42.99	77.59	73.27	55.56	93.00	85.17
L1SVM	74.44	79.20	52.78	81.39	74.41	59.78	94.86	88.85
GSR	68.99	79.58	73.59	81.52	83.41	68.10	83.02	85.60
LGS	69.48	91.12	77.49	82.24	93.57	82.01	81.98	90.57
MKVM	83.15	69.98	69.11	87.68	72.91	77.64	85.21	89.51
CSL	79.62	93.03	54.44	**94.75**	75.26	67.72	83.70	69.74
SRSL	85.94	76.35	83.33	93.33	93.08	**86.15**	90.63	88.97
RMM	85.95	70.28	71.67	93.87	72.39	81.02	86.42	91.41
Proposed	**88.44**	**98.59**	**89.96**	92.96	**98.99**	80.99	**99.01**	**96.99**

consider the correlation between the samples so that the graph methods achieve better performance than other methods. Third, noise seriously affects the classification results of all methods. For example, the data sets with less noise (*i.e.,* Synb1 and Synm1) outperform the data sets with more noise (*i.e.,* Synb1 and Synm2), in terms of classification results on all methods.

We conduct experiments with all methods on real data sets to output a part of all the features (*i.e.,* 10%, 30%, 50%, 70% and 90%) to evaluate the classification results of multi-class classification and binary classification, respectively. We list the classification accuracy of multi-class classification of all methods with all features and different ratios of features, respectively, in Table 3.4 and Figure 3.1. We also report the classification accuracy of binary classification of all methods with all features and different ratios of features, respectively, in Table 3.5 and Figure 3.2.

In multi-class classification, firstly, the proposed method achieves the best classification performance, followed by SRSL, LGS, RMM, MKVM, GSR, CSL, L1SVM and Baseline. For example, the proposed method improved by 6.02% and 17.53% on average, with respect to classification accuracy(ACC), compared with the best method (*i.e.,* SRSL) and the worst comparison method (*i.e.,* L1SVM) except the Baseline method. The reason is that our method utilizes both dynamic graph structure and the self-paced learning together. In detail, there are two methods (*i.e.,* GSR and LGS) consider one of the former technologies and there are three methods



(a) Wavef (b) Lung (c) Ecoli (d) Cane

(e) Warpar (f) Yale32 (g) Coil (h) Isolet

Figure 3.1: Classification accuracy (ACC) of all methods at different ratios of selected features.

Table 3.5: The classification results (%) of all methods on binary classification.

Data Sets	Parkin				Spambase				Hillv				German			
	ACC	SEN	SPE	AUC	ACC	SEN	SPE	AUC	ACC	SEN	SPE	AUC	ACC	SEN	SPE	AUC
Baseline	56.06	38.27	73.85	72.06	60.60	52.14	63.57	62.47	62.40	57.50	60.52	42.20	70.00	72.50	75.00	66.25
L1SVM	59.75	45.77	81.73	78.92	60.58	53.71	64.76	69.72	66.54	68.11	79.21	70.99	70.02	44.89	72.54	67.57
GSR	47.98	31.73	64.23	86.53	69.98	58.88	68.11	73.54	74.64	71.14	75.47	77.81	69.94	41.57	72.42	81.43
LGS	53.94	75.96	71.92	88.84	77.89	79.22	72.05	78.67	74.76	72.13	79.23	**80.37**	70.00	**79.99**	75.17	72.57
MKVM	53.41	75.14	69.47	63.00	59.87	77.48	69.42	68.33	69.84	72.58	72.41	69.80	70.08	65.00	68.91	59.97
CSL	64.88	59.90	62.68	58.88	67.02	67.48	71.02	66.57	64.81	63.58	73.69	68.71	59.51	59.47	65.42	67.15
SRSL	62.50	67.40	63.17	61.08	91.85	79.68	72.11	73.43	65.51	69.90	72.47	67.00	**76.20**	73.10	68.44	66.50
RMM	61.06	77.12	70.50	65.15	61.17	81.01	71.44	72.09	70.49	**78.79**	70.51	72.09	70.07	65.96	71.16	62.09
Proposed	**67.79**	**78.10**	**82.22**	**90.43**	**93.56**	**85.54**	**73.22**	**80.59**	**75.67**	73.31	**84.74**	76.84	75.60	75.50	**80.57**	**82.35**

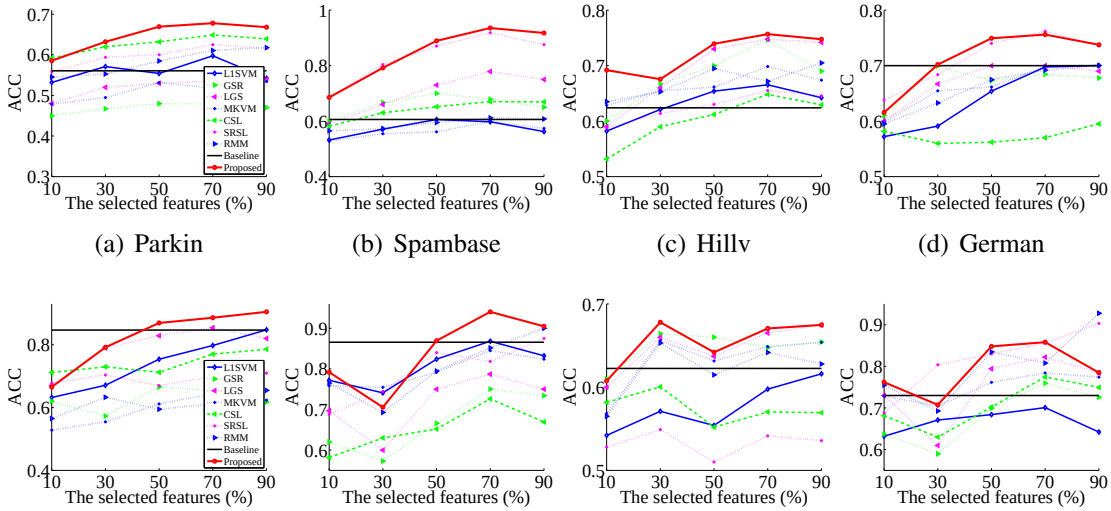
Data Sets	Colon				Dbworld				Madelon				Pcmac			
	ACC	SEN	SPE	AUC	ACC	SEN	SPE	AUC	ACC	SEN	SPE	AUC	ACC	SEN	SPE	AUC
Baseline	84.67	80.00	76.67	78.76	86.57	86.67	85.50	84.81	62.25	62.90	61.60	37.41	73.00	76.33	69.17	61.84
L1SVM	84.76	68.51	72.54	72.50	86.81	85.13	85.43	83.33	61.60	60.80	62.40	48.14	70.08	77.76	71.15	62.94
GSR	66.57	73.41	78.19	63.75	75.00	70.40	81.91	69.03	66.43	73.41	75.90	69.99	75.93	75.82	71.20	66.07
LGS	85.43	83.41	87.35	76.49	78.69	69.87	68.54	73.91	67.45	61.30	**77.60**	69.19	82.21	69.77	75.98	67.40
MKVM	64.44	73.41	74.05	68.88	84.49	69.47	68.09	69.71	65.60	63.57	69.70	58.79	78.41	69.58	72.23	62.87
CSL	78.57	67.99	78.12	74.00	72.62	59.99	73.41	70.04	60.04	61.70	63.47	67.90	77.47	76.12	**77.81**	**70.58**
SRSL	70.97	68.42	69.07	69.30	87.50	74.68	76.15	67.30	54.90	49.99	60.01	58.47	90.32	**78.08**	71.09	67.00
RMM	65.48	75.00	75.07	72.09	90.00	73.33	77.50	72.09	65.30	**76.00**	74.60	52.09	**92.75**	76.29	72.19	62.00
Proposed	**90.48**	**84.30**	**91.18**	**80.43**	**94.05**	**87.54**	**85.58**	**86.18**	**67.79**	71.31	74.74	**73.37**	85.81	75.70	69.48	67.17

(*i.e.,* MKVM, CSL, and RMM) utilize other approaches to improve the SVM. Secondly, Table 3.4 indicate that it may not lead to better result by considering a single technique. Moreover, different ratios of features reveal diverse results with regard to classification in Figure 3.1. However, there are comparison methods produce worse results than the Baseline which only use the conventional SVM. The reason is that all features may provide more comprehensive information from the original data. Thirdly, two linear SVM (including L1SVM and Baseline) get the worse

results than other comparison methods since these two methods are designed to deal with the classical binary classification instead of the multi-class classification.

Similar to the results of multi-class classification, the proposed method obtain the best classification performance in term of four evaluation metrics compared with comparison methods. Specifically, as for the classification accuracy(ACC), the proposed method obtain the best performance with five data sets out of eight data sets, followed by SRSL, LGS, RMM, L1SVM, Baseline, GSR, MKVM, and CSL. For example, the proposed method averagely improved by 6.38%, 7.55%, 9.21%, 9.30%, 11.33%, 11.9%, 13.04%, 13.08%, and 13.23% respectively, compared to SRSL, LGS, RMM, L1SVM, Baseline, GSR, MKVM, and CSL.

Besides, the multi-class methods with other techniques occupy two of the top four among the comparison methods, and the linear SVM model (*i.e.,* L1SVM and Baseline) is better than the conventional feature selection methods with the linear kernel function. Moreover, in terms of sensitivity(SEN), specificity(SPE), and Area Under Curve (AUC), compared to other eleven comparison methods, the proposed method still could achieve the best results in most of the data sets. This contributes to the fact that the proposed method utilizes two methods to remove outliers of the original data in the low-dimensional subspace and employ the robust loss function to conduct feature selection. On the contrary, other comparison methods may consider one aspect or use other technology to improve the basic SVM model.



(a) Parkin (b) Spambase (c) Hillv (d) German

(e) Colon (f) Dbworld (g) Madelon (h) Pcmac

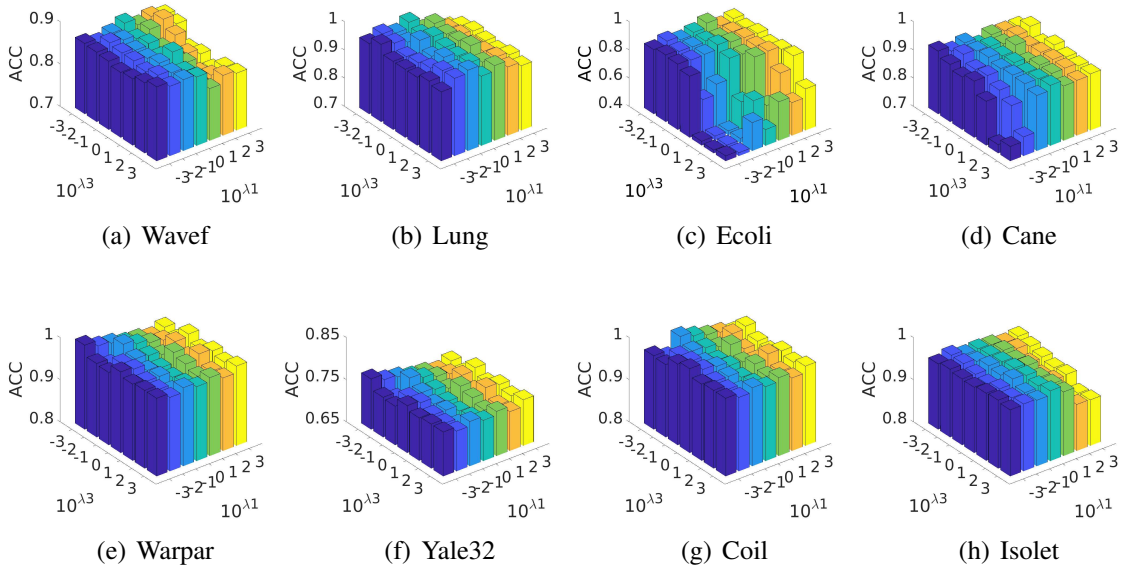
Figure 3.2: Classification accuracy (ACC) of all methods at different ratios of selected features for binary classification.

3.5 Discussion

3.5.1 Parameters sensitivity analysis

Although our objective function in Eq. (3.6) has four parameters to be tuned, there are two parameters can be adjusted automatically, *i.e.,* γ and λ_2. Hence, we tune the parameters λ_1 and λ_3 within the range of $\{10^{(-3)}, 10^{(-2)}, ..., 10^3\}$. Meanwhile, we investigate the variations of the classification accuracy of our method while keeping the left features as 70% of all the features and the results on sixteen data sets are showed in Figures 3.3-3.4.

For example, in Figure 3.3, we can find that our method achieves the best performance on the data sets Cane and Coil while setting $\lambda_1 = 10$, and $\lambda_3 = 0.1$. Meanwhile, our method produces the best ACC 99.01% on Coil for multi-classification data sets. Moreover, for binary classification data sets in Figure 3.4, the best ACC 94.05% of our method can be found on Dbworld data set. Furthermore, our method has a higher probability of obtaining the best performance on all binary classification data sets while $\lambda_3 = 10^3$, such as Parkin, German, Dbworld, and Madelon.



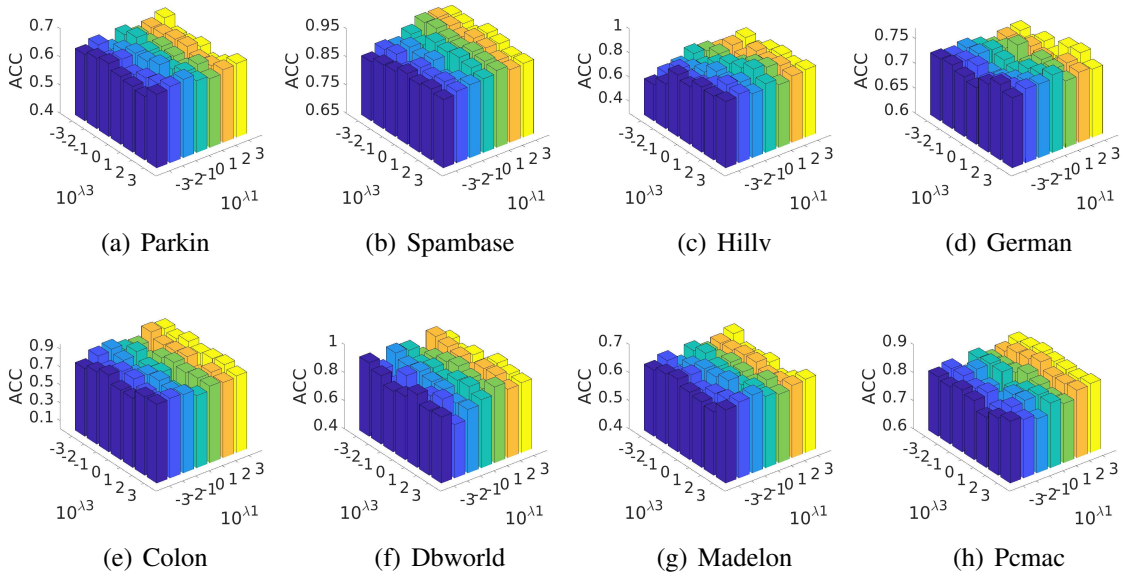
(a) Wavef (b) Lung (c) Ecoli (d) Cane

(e) Warpar (f) Yale32 (g) Coil (h) Isolet

Figure 3.3: Classification accuracy (ACC) of the proposed method at different parameters' setting on the variables $\lambda 1$ and $\lambda 3$ for multi-class classification.

3.5.2 Convergence

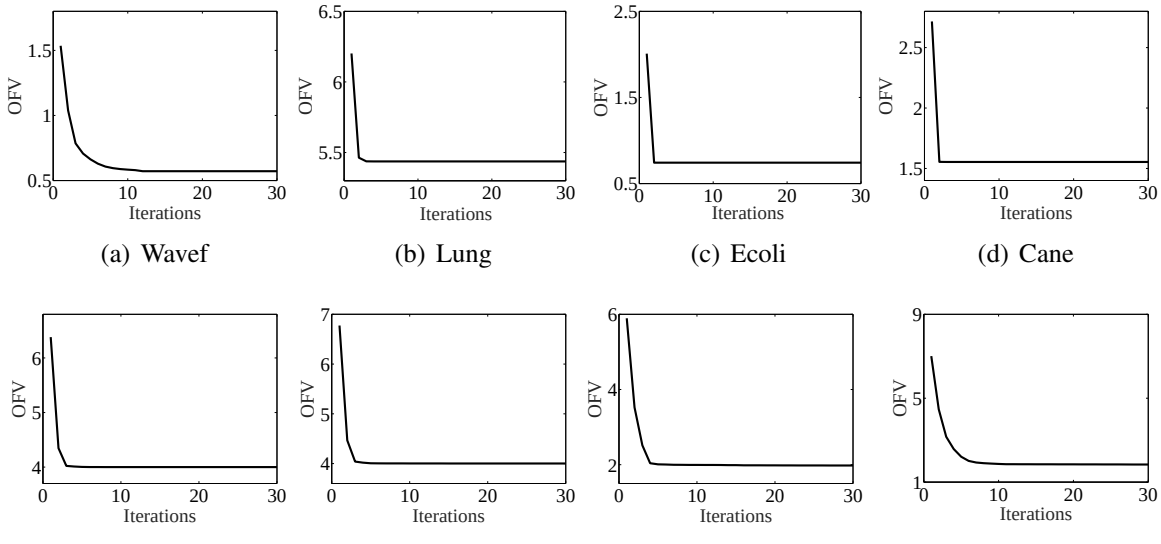
We propose a new method to optimize our proposed objective function Eq. (3.6) and theoretically prove its convergence. We experimentally verify the convergence of objective function by investigating the variations of the objective function values of Eq. (3.6) at different iterations. We set the stop criteria of our algorithm as $\frac{\|obj(t+1)-obj(t)\|_2^2}{obj(t)} \leq 10^{-5}$, where $obj(t)$ represents the value of objective function Eq. (3.6) at the t-th iteration.



(a) Parkin (b) Spambase (c) Hillv (d) German

(e) Colon (f) Dbworld (g) Madelon (h) Pcmac

Figure 3.4: Classification accuracy (ACC) of the proposed method at different parameters' setting on the variables $\lambda 1$ and $\lambda 3$ for binary classification.



(a) Wavef (b) Lung (c) Ecoli (d) Cane

(e) Warpar (f) Yale32 (g) Coil (h) Isolet

Figure 3.5: Objective Function Values (OFV) of the proposed method at different iterations for multi-class classification.

In Figures 3.5-3.6, we have at least two observations: (1) the proposed algorithm sharply decrease the objective function values in the first several iterations and then begin to stabilise; and (2) objective function converges within tens iterations on all the data sets. These conclusions indicate

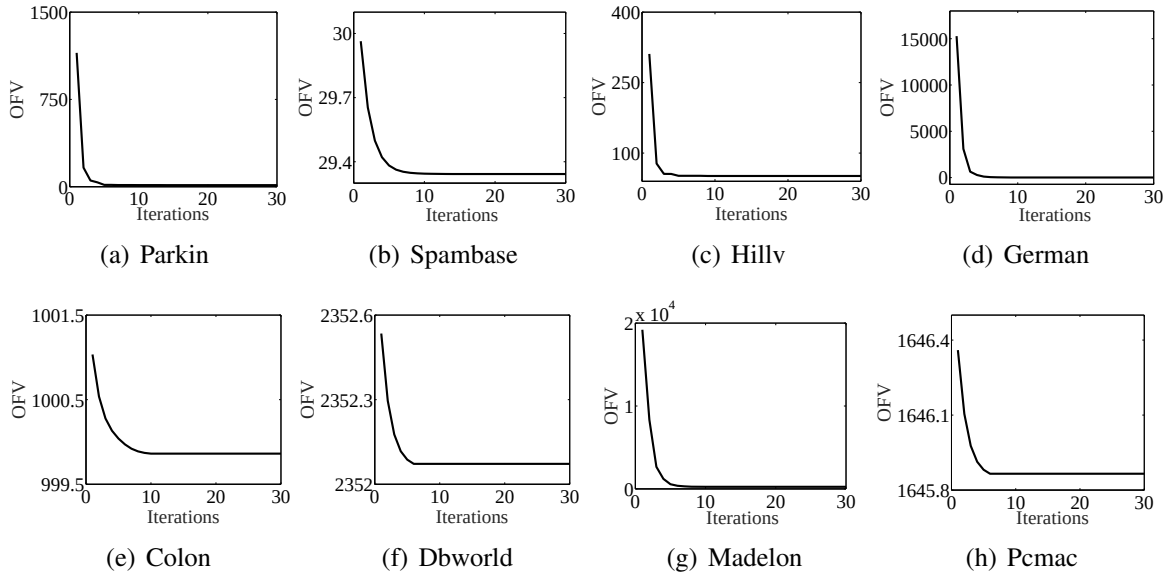


Figure 3.6: Objective Function Values (OFV) of the proposed method at different iterations for binary class classification.

that our method have solved the proposed objective function in Eq. (3.6) and achieved fast convergence.

3.6 Summary

This chapter has presented a new robust SVM classifier via integrating the hinge loss function, a self-paced learning, the graph learning with an $\ell_{2,1}$-norm regularizer into a unified framework to simultaneously learn both important samples and features in the robust low-dimensional subspace. In this way, it solved the issues of the robustness and the interpretability of graph learning under the supervised learning. Experimental results on both synthetic data sets and real data sets verified that our proposed method achieved the best classification performance, compared to the state-of-the-art classification methods.

This chapter has been published in the CORE rank A journal, *i.e.,* World Wide Web journal [67].

Chapter 4

Multi-band brain network analysis for functional neuroimaging biomarker identification

4.1 Introduction

Resting state functional Magnetic Resonance Imaging (fMRI) has been verified to have the potential of improving neuro-disease diagnosis by constructing Functionally Connectivity brain Networks (FCNs) [55, 79]. It results in a comprehensive understanding of neurological disorders at a whole-brain level by measuring synchronized time-dependent changes in the Blood Oxygenation Level Dependent (BOLD) signals [57, 90]. Hence, fMRI has been becoming a valuable technique for identifying biomarkers with neuroimaging data.

Correlation-based methods are commonly used to construct fully connected FCNs, where each node (*i.e.,* one brain region) connects with all nodes and each edge measures the synchronization degree of functional activities [133, 144]. Traditional Pearson correlation analysis only captures pairwise information and thus is vulnerable to spurious or insignificant functional connectivities. Recently, sparse methods [89, 145, 203] were proposed to construct sparsely connected FCNs (sparse FCN for short), where each node connects with a part of nodes to reduce the influence of unreal or unimportant functional connections. However, previous methods for neuro-disease analysis on fMRI data, such as Pearson correlation analysis and sparse methods, still have to face many challenges due to the reasons, including heterogeneity across subjects, the curse of dimensionality, noise influence, inter-subject variability, *etc.*

In search of significant disease diagnosis, there is a consensus that BOLD signal only contains a small portion of frequencies (*i.e.,* 0.01HZ-0.08HZ) that are related to neural activity [160, 176]. Based on this observation of fMRI, current computational methods characterize the full connectivity using the filtered BOLD signals where some frequency bands of the signals have been filtered, with the assumption that the filtered BOLD signals can reflect the complex brain func-

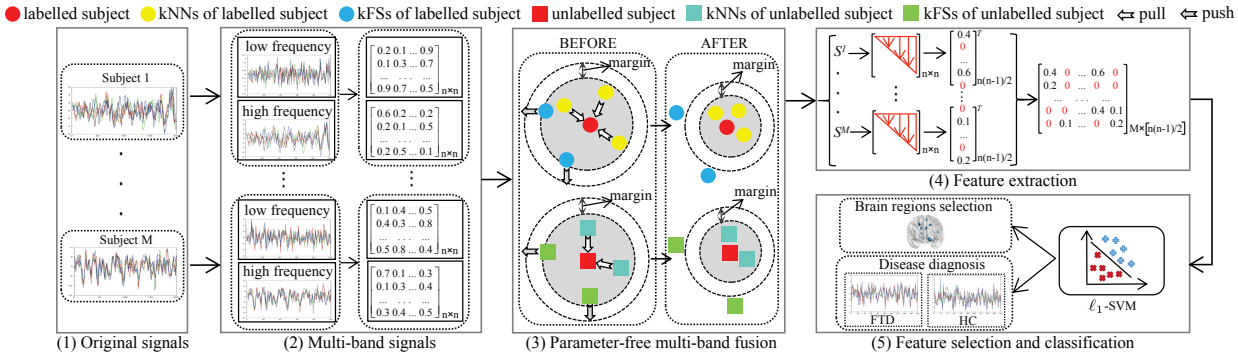


Figure 4.1: The proposed framework for functional neuroimaging biomarker identification: (1) Original Blood Oxygenation Level Dependent (BOLD) signals for M subjects; (2) Each signal was first partitioned into multi-band signals (*e.g.,* a low frequency signal and a high frequency signal) using the Discrete Wavelet Transform (DWT) method and the Pearson correlation coefficient was then calculated on each frequency band signal to obtain the fully connected Functionally Connectivity brain Networks (FCNs); (3) The proposed parameter-free multi-band fusion model is designed to automatically learn a sparse FCN $\mathbf{S}^m$ $(m = 1, ..., M)$ by fusing multiple fully connected FCNs for each subject, as well as to learn sparse FCNs for all subjects by pulling each sparse FCN to be close to its k nearest neighbors (kNNs) and pushing each subject to be far away from its k furthest sparse FCNs (kFSs); (4) A data matrix $\mathbf{X}$ is obtained by extracting the upper triangle part of $\mathbf{S}^m$; (5) The ℓ_1-SVM is employed to jointly construct feature selection (*i.e.,* the connection between two brain regions) and the classification task (*i.e.,* disease diagnosis).

tions for all brain regions [50]. As a complex system, however, each brain region consists of functionally similar neurons that support differentiable functions. In this regard, one possible solution would be fine-tuning the BOLD signals into a fine-grained frequency band tailored to the brain function of subspecialized brain regions. Since the exact definition of brain function in each region is still largely under debate [161], the alternative solution is to disentangle the heterogeneity in BOLD signals. Specifically, the BOLD signals are first decomposed into multiple frequency bands (multi-band for short) based on the characteristics of full connectivity. The multi-band signals are then fused into a unified region with adaptively full connectivity representation, which can significantly enhance the diagnostic power of connectivity biomarkers for the computer-assisted diagnosis.

After obtaining multiple fully connected FCNs for each subject, it is necessary to combine them into a common fully connected or sparse FCN, aiming at learning the most representative features across individuals (or subjects). However, different frequency bands of neuronal signals usually carry unique functional information, which is different from others to support differentiable functions [161], *i.e.,* band diversity for short. Hence, it is unreasonable to construct a common FCN by averaging different frequency bands for each subject. Moreover, it is difficult to align FCNs across subjects due to the heterogeneity across subjects.

In this chapter, we hypothesize that the observed brain activity is a mixture of harmonic signals with different frequency bands and the dysfunction patterns of different brain disorders have

different responses in different frequency bands, and thus propose a functional connectivity analysis framework to jointly conduct feature learning and personalized disease diagnosis with fMRI data in a semi-supervised manner. Specifically, the proposed framework is listed in Figure 4.1 by involving the following key steps. (1) We first apply the wavelet toward the mean time courses on each brain region to obtain the wavelet coefficient at each frequency, and further employ the conventional Pearson correlation analysis to obtain a fully connected FCN for each subject at the underlying frequency. (2) We investigate a parameter-free multi-band fusion method to automatically output a sparse FCN by fusing multiple fully connected FCNs for each subject, as well as to let each sparse FCN be close to its near sparse FCNs and be far away from its furthest sparse FCNs. (3) We employ the ℓ_1-SVM to jointly select brain regions (*i.e.,* feature selection) and conduct disease diagnosis (*i.e.,* classification).

4.2 Related work

The fMRI is primarily used for the study of early disease diagnosis since it can complement other brain physiological indicators and can integrate a large number of biomarkers with BOLD signals [96]. Figure 4.2 indicates four steps in the framework of medical image analysis. (1) Data pre-processing. The original signals are obtained from fMRI, and it can be dealt with by the steps, including the sliding window, removing part of data with experience, and multi-band method. (2) Correlation matrix generation. Every similarity matrix is acquired by the various methods, *i.e.,* Pearson correlation coefficient (PCC), sparse learning method and graph-based method. (3) Feature learning. The feature of each vector is obtained by the diverse methods, *i.e.,* dictionary learning method and the method extracting the upper triangular matrix to represent a subject with a vector. (4) Disease diagnosis. The task of diagnosis is conducted by the combination of all vectors through the specific classifier *i.e.,* support vector machine (SVM).

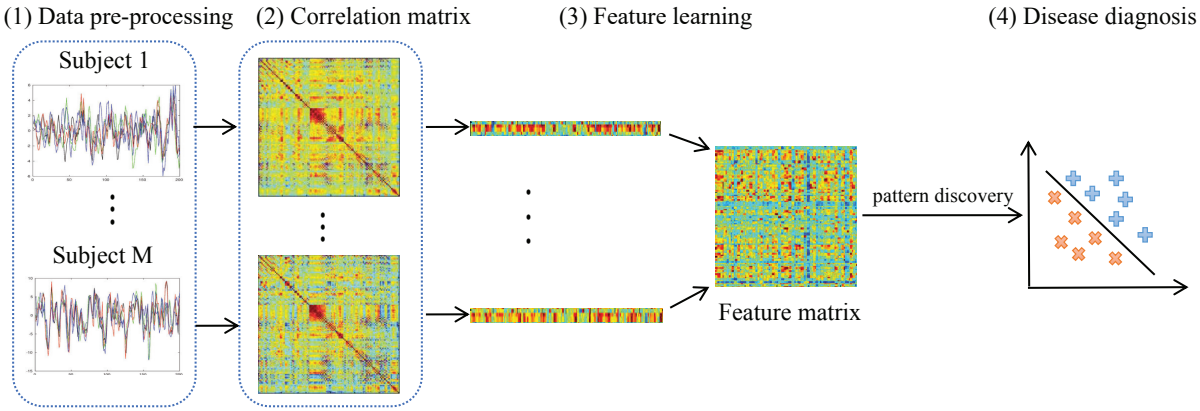


Figure 4.2: The framework of medical image analysis for disease diagnosis.

4.2.1 Data pre-processing

There are four traditional methods to deal with the BOLD signals for the fMRI analysis. First, the original signals is directly used for the next steps. For example, Zheng *et al.* propose the intensity transformation to model the enhancement map work on the original MRI data [212]. Shi *et al.* utilize specific subject atlas to guide neonatal image segmentation for the original data [143]. Second, the sliding window method partitions the signals into a slice of overlapping segments [21, 23, 105, 202]. For example, Shakil *et al.* propose the adaptive window analysis method [138] and Chen *et al.* sequentially utilize the sliding window method and the agglomerative clustering [22]. Third, unnecessary signals are removed with experience by assuming that the selected signals data between the restricted interval is useful for better explanation. For example, Macey *et al.* remove the first volume to prevent the influence of large global BOLD signals [113]. Jarmasz *et al.* select the specific time series to meet the criterion of certain statistic [73]. Last, multiple bands methods employ the band-pass filter to divide the original signals into several bands with frequency-specific. For example, Gao *et al.* obtain various band-pass dependence by using discrete wavelet transform [44] and Zhang *et al.* obtain the sub-band signals from the original signals [204].

Deep learning methods have been used to deal with the original BOLD signals. For example, Nguyen *et al.* synthesize new fMRI data by the co-registration data augmentation method [121]. Huang *et al.* propose to reconstruct each original fMRI data [68]. Seeliger *et al.* propose to explore the abilities of reconstructing arbitrary fMRI data [137].

4.2.2 Correlation matrix generation

The methods of correlation matrix generation include Pearson correlation coefficient methods, sparse methods, and graph methods.

Pearson correlation coefficient method is used to obtain the fully connected FCNs in the brain data analysis due to its ability for the strength of the linear correlation between individual pairs [3, 202, 204]. For example, Liao *et al.* employ it to explore the correlations between the value of regional homogeneity and clinical results in the patient group and calculate the correlations of scores between the spastic paraplegia rating scale and functional connectivity [104]. Zhang *et al.* utilize it to obtain the correlations in the low-order functional connectivity network [202].

Sparse methods not only select the important correlations among the subjects for obtaining the sparse-connected FCNs, but also obtain the sparse-coding dictionary for predicting the unseen subject [160]. For example, Zhang *et al.* compare three variants of sparse methods [198] and Yu *et al.* propose to consider the link information, group structure, and sparse representation simultaneously to calculate the correlations [180].

Graph methods decompose the functional structure into the combination of nodes (*i.e.,* the subjects) and edges (*i.e.,* the correlations between any subjects) [179]. For example, Yuan *et al.* propose to represent the spatio-temporal dynamic interactions of brain functional networks [183].

Recently, deep learning methods are designed to find the correlations between the brain regions.

For example, Suk *et al.* propose a deep auto-encoder model to search the hierarchical non-linear functional correlations among the brain regions [148]. Sarraf *et al.* employ the LeNet-5 to fit the fMRI data for Alzheimer disease classification [135]. Zhang *et al.* propose to extract the informative features from complex neuroimaging data through a hierarchical way [199].

4.2.3 Feature learning

Feature learning not only finds the correlations between brain regions, but also extracts the important features (*i.e.,* the connections between two brain regions) to represent the original similarity matrix. Popular methods have the methods of using the original similarity matrix directly, the methods of extracting the feature vector from the original similarity matrix, the methods of using the dictionary learning, and deep methods.

First, the original similarity matrix is used to represent individual brain networks in the brain graph. Kong *et al.* set the specific sparsity threshold to cope with the similarity matrix, and directly apply the similarity matrix to draw the individual brain network for MRI data [91]. Angulakshmi *et al.* employ the spectral clustering work on the constructed similarity matrix to serve the tumour segmentation [4]. Second, the upper triangular is extracted from the similarity matrix. For example, Khavari *et al.* vectorize the upper triangular matrix to minimize the effects of dimensional complexity [85]. Third, the dictionary learning method is designed to find a set of basic elements (*i.e.,* the dictionary) so that the input data has a sparse representation mapping to this set of basic elements. For example, Chen *et al.* employ the weighted-graph local clustering to extract the feature vector from the correlation between mean correlation time series of clusters [21]. Liu *et al.* employ the sparse method to conduct the feature selection from the diverse feature matrix [105]. Last, deep methods are employed to extract the similarity vector. For example, Ktena *et al.* employ the siamese graph convolutional neural network to select the similarity vector between any two subjects [93]. Yao *et al.* employ multi-scale templates to obtain multiple functional connection networks [177].

4.2.4 Disease diagnosis

Disease diagnosis employs or designs classifiers to conduct disease diagnosis. Support Vector Machine classifier (SVM) is the classical and effective classifier in medical image analysis [21–23]. For example, Zhang *et al.* employ it for the classification tasks under the multi-kernel learning manner [202]. Liu *et al.* employ SVM to examine both low-order and high-order connections [105]. Deep learning methods usually conduct classification task (*i.e.,* diagnosis) by fully connected (FC) layers. For example, Ma *et al.* employ the fully connected layer to validate the similarity learning [112]. Yuan *et al.* utilize FC layer as the input for the classifier (*i.e.,* softmax activation function) [184].

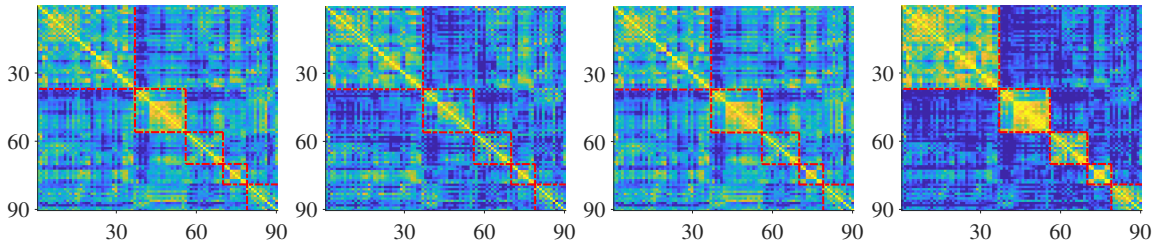


Figure 4.3: Visualization of correlation analysis of a signal from the data set FTD with different frequency bands, *i.e.,* the original signal, the low frequency band signal, the high frequency band signal, and the multi-band signal, from left to right.

4.3 Method

Assume we have *M* subjects and each subject has the BOLD signal $\mathbf{B}^m \in \mathbb{R}^{n\times t}$ $(m = 1, ..., M)$ where *n* and *t*, respectively, represent the number of brain regions and the length of signals, we denote $\mathbf{A}^{m,v} \in \mathbb{R}^{n\times n}$ $(v = 1, ..., V)$ as the fully connected FCN of the *v*-th band signal of the *m*-th subject obtained by Pearson correlation analysis on *n* brain regions, this work investigates to learn a common sparse FCN $\mathbf{S}^m \in \mathbb{R}^{n\times n}$ for each subject so that $\mathbf{S}^m$ could fuse the functional connectivity from *V* fully connected FCNs $\mathbf{A}^{m,v}$, as well as is close to its neighbors and is far away from its furthest sparse FCNs. As a result, it is homogeneous to other sparse FCNs.

4.3.1 Multi-band signals

Conventional methods of functional connectivity using fMRI data focused on characterizing BOLD signals with low frequency range (generally from 0.01HZ to 0.08HZ) [58, 82, 144]. In particular, the signals within the frequency range close to 0.00HZ are more likely to be affected by the periodically noisy signals generated by the undersampled periodic hardware [82, 89], the signals within the frequency range in 0.00-0.01HZ are significantly related to non-physiologic origin (*i.e.,* the MRI scanner drift) and are usually treated as covariates of no interest in the statistical analysis [37]. Therefore, the removal of the signals within the frequency range of [0.00HZ, 0.01HZ] ensures that the obtained BOLD signals are predominately related to the physiological state. Furthermore, the removal of the signals above 0.08HZ can minimize the interference of other external signals [110]. Recently, studies demonstrated that the dysfunctional patterns affecting the neuro-diseases are the mixture of brain activity with multiple frequency bands [161]. Due to the complexity of human brains, the frequency band usually dominates the subtle functional patterns that are specific to certain disorders. Even under the resting state, the complexity of brain activity is beyond the power of single frequency band [152]. In addition, the changes across different frequency bands (*i.e.,* band diversity) have been known little [50, 153]. Hence, it is challenging to consider multiple frequency bands for neuro-disease analysis with fMRI data.

4.3.2 Sparse FCN learning

We employ Pearson correlation analysis to obtain V fully connected FCNs using multi-band signals. However, there are at least two issues that need to be addressed, *i.e.,* band diversity and FCN's interpretability.

First, band diversity indicates that different band signals contain different characteristics, as shown in Figure 4.3, where either the low frequency band signal or the high frequency band signal has different information, compared to the original signal which is a mixture of the low frequency band signal and the high frequency band signal. Moreover, the low frequency band and the high frequency band have complementary information to each other. For example, the solid yellow rectangle in the low frequency band signal (*i.e.,* the second red dot rectangle) is unclear as well as small, while the solid yellow rectangle in the high frequency band signal is clear as well as big, similar to the original signal in Figure 4.3. Both the difference and the complementary between the low frequency band signals and the high frequency band signals motivate us to first decompose them and then to fuse them, as shown in Figure 4.3, where our proposed multi-band signal removes the noise (*i.e.,* correlations are far away from the diagonal) as well as clearly preserves the local structures (*i.e.,* five red dot rectangles across the diagonal).

The second drawback of the fully connected FCN is the lack of interpretability. Moreover, its connectivity may contain noise (*e.g.,* either irrelevant or spurious connectivities) to possibly affect the analysis of brain networks [131,167]. Neurologically, a certain brain activity or a specific disease predominantly interacted with a part of brain regions. Therefore, the sparse connectivities are preferred to construct brain connectivity networks [203].

Given multiple fully connected FCNs for an individual subject, by considering the band diversity of every signal and the interpretability of every fully connected FCN, in this chapter, we investigate to fuse the fully connected FCNs of every subject to a sparse FCN by

$$\begin{aligned} \min_{\mathbf{S}^1,...,\mathbf{S}^M} & \sum_{m=1}^{M}\sum_{v=1}^{V}||\mathbf{S}^m - \mathbf{A}^{m,v}||_F^2 + \alpha\mathcal{R}(\mathbf{S}^1,...,\mathbf{S}^M) \\ & s.t.,\ \forall i, \mathbf{s}_{i,\cdot}^{m^T}\mathbf{1} = 1, s_{i,i}^m = 0, s_{i,j}^m \geq 0\ if\ j \in \mathcal{N}(i),\ otherwise\ 0, \end{aligned} \tag{4.1}$$

where $\|\cdot\|_F$ represents the Frobenius norm and α is a non-negative tuning parameter. $\mathcal{R}(\mathbf{S}^1,...,\mathbf{S}^M)$ is the penalty or constraint on $\mathbf{S}^m$ $(m = 1,...,M)$. $\mathbf{s}_{i,\cdot}^{m^T}$ and $s_{i,j}^m$, respectively, represent the *i*-the row of $\mathbf{S}^m$ and the *i*-th row and the *j*-th column element of $\mathbf{S}^m$. $\mathbf{1}$ and $\mathcal{N}(i)$, respectively, indicate the all-one-element vector and the nearest neighbor set of the i-th brain region. The constraint "$\mathbf{s}_{i,\cdot}^{m^T}\mathbf{1} = 1, s_{i,i}^m = 0, s_{i,j}^m \geq 0\ if\ j \in \mathcal{N}(i),\ otherwise\ 0$" implies that each node $s_{i,\cdot}^m$ is sparsely represented by other nodes $s_{j,\cdot}^m$ $(i \neq j, i, j = 1,...,n)$.

Compared to previous methods, Eq. (4.1) has the following advantages. First, Eq. (4.1) aims at obtaining a sparse FCN $\mathbf{S}^m$ for the *m*-th subject based on V fully connected FCNs $\mathbf{A}^{m,v}$ $(v = 1,...,V)$ by considering the band diversity. Second, the new representation $\mathbf{S}^m$ is iteratively updated by Eq. (4.1). Specifically, the value of $\mathbf{S}^m$ can be adjusted with the updated $\mathbf{S}^{m'}$ $(m \neq m')$. It is noteworthy that previous methods (*e.g.,* sparse coding [33] and sparse graph [203]) generate unchanged FCNs. Third, the sparse number for each row of $\mathbf{S}^m$ varies based on the data

distribution. Specifically, the connectivity number (*i.e.,* the non-zero number for each row) for each node is automatically decided by Lemma 5 in Section 4.3.5.

Finally, we list our motivation of fusing information across different frequency bands as well as different subjects in Eq. (4.1) as follows.

First, different frequency bands of the signals contain different important information and noise, so it is intuitive and popular to combine multi-source data (*i.e.,* the information across different frequency bands in this work) together to obtain discriminative representations in the domains of machine learning and medical image analysis. To achieve this, Eq. (4.1) learns a common sparse representation $\mathbf{S}^m$ $(m = 1, ..., M)$ for each subject by fusing multiple fully connected FCNs $\mathbf{A}^{m,v}$ $(v = 1, ..., V)$ collected from different frequency bands.

Second, if the representation of each subject is obtained independently, the obtained representation $\mathbf{S}^m$ for the m-th subject is easily heterogeneous to other representations $\mathbf{S}^{m'}$ $(m \neq m')$. Eq. (4.1) learns the representations of all subjects in the same framework by using the regularization term $\mathcal{R}(\mathbf{S}^1, ..., \mathbf{S}^M)$ defined in Section 4.3.3, aiming at learning homogenous representations for all subjects. Moreover, minimizing the error across all subjects is to learn a common sparsely connected representation $\mathbf{S}^m$ for each subject as well as to generate homogeneous representations for all subjects. In the literature, minimizing the error across all subjects is popular. For example, Hinrich *et al.* proposed minimizing the error of negative log-likelihood across subjects to ensure the temporal components from a consistent set of brain regions [62].

As a result, Eq. (4.1) considers the fusion across frequency bands as well as all subjects simultaneously to output discriminative and homogenous representations for all subjects. Physiologically, the fusion in Eq. (4.1) can explore the discriminative information implicitly across frequency bands [200], and can enhance data consistency across subjects [19].

4.3.3 Parameter-free multi-band fusion

Without taking the constraint $\mathcal{R}(\mathbf{S}^1, ..., \mathbf{S}^M)$ into account, the optimization of $\mathbf{S}^m$ is independent on the optimization of $\mathbf{S}^{m'}$ $(m \neq m')$. This may output trivial solutions for $\mathbf{S}^m$, *i.e.,* the average of $\mathbf{A}^{m,v}$ $(v = 1, ..., V)$. To solve this issue, we define the constraint $\mathcal{R}(\mathbf{S}^1, ..., \mathbf{S}^M)$ in Eq. (4.1) based on the following observations.

First, in real applications, the BOLD signals usually come from different places such as different hospitals and different machines, so the fully connected FCNs are easily heterogeneous to each other, *i.e.,* heterogeneity across subjects. It is straightforward to smooth all FCNs so that they are homogeneous. Second, in our proposed personalized classifier, the training set includes labelled subjects and unlabelled subjects. Specifically, given a test subject, the personalized framework makes full use of all labelled subjects and unlabelled subjects to construct the learning model, and thus it is exactly a semi-supervised manner. Hence, it is very helpful if the outputted $\mathbf{S}^m$ has significant discriminative ability.

Weinberger and Saul proposed a supervised Large Margin Nearest Neighbor (LMNN) to conduct metric learning by keeping the local neighborhood of the training subjects [165], *i.e.,* the neigh-

bors of each subject in the new feature space are exactly its original neighbors in the original feature space. Specifically, the first term of the LMNN penalizes large distances between each subject and its original neighbors with the same label, and its second term penalizes the small distances of the subjects with different labels. As a result, the labelled subjects are close to the subjects with the same label and are far away from the subjects with different labels. In this way, the LMNN classifier has discriminative ability. However, LMNN was designed for metric learning and did not consider the discriminative ability of unlabelled subjects.

In this work, considering the semi-supervised scenario where the training subjects include labelled subjects and unlabelled subjects, we first have an observation that the *k* nearest neighbor (*k*NN) classifier always classifies subjects to the class of their nearest neighbors. Hence, each subject (either a labelled subject or an unlabelled subject) should share the same label with its *k*NNs and should have different labels to its distant subjects. More specifically, by denoting "*k* nearest neighbors" and "*k* furthest subjects", respectively, as "*k*NNs" and "*k*FSs", the set of labelled subjects and the set of unlabelled subjects, as $\mathcal{L}$ and $\mathcal{U}$, respectively, we define the neighbor set $\mathcal{N}(i)$ and the distant set $\mathcal{F}(i)$ as follows:

Definition 1. *$\mathcal{N}(i)$ of the i-th unlabelled subject includes its kNNs in $\mathcal{L}\bigcup\mathcal{U}$, and $\mathcal{N}(i)$ of the i-th labelled subject includes its kNNs with the same label in $\mathcal{L}$,* i.e.,

$$\mathcal{N}(i) = \begin{cases} kNNs \ in \ \mathcal{L}\bigcup\mathcal{U}, \ i \in \mathcal{U} \\ kNNs \ with \ the \ same \ label \ in \ \mathcal{L}, \ i \in \mathcal{L} \end{cases} \tag{4.2}$$

Definition 2. *$\mathcal{F}(i)$ of the i-th unlabelled subject includes its kFSs in $\mathcal{L}\bigcup\mathcal{U}$, and $\mathcal{F}(i)$ of the i-th labelled subject includes its kFSs with different labels in $\mathcal{L}$,* i.e.,

$$\mathcal{F}(i) = \begin{cases} kFSs \ in \ \mathcal{L}\bigcup\mathcal{U}, \ i \in \mathcal{U} \\ kFSs \ with \ different \ labels \ in \ \mathcal{L}, \ i \in \mathcal{L} \end{cases} \tag{4.3}$$

We then define $\mathcal{R}(\mathbf{S}^1, ..., \mathbf{S}^M)$ as

$$\begin{aligned} &\sum_{m=1}^{M}(\sum_{p\in\mathcal{N}(m)} ||\mathbf{S}^m - \mathbf{S}^p||_F^2 \\ &+\beta[1 + \sum_{p\in\mathcal{N}(m)} ||\mathbf{S}^m - \mathbf{S}^p||_F^2 - \sum_{q\in\mathcal{F}(m)} ||\mathbf{S}^m - \mathbf{S}^q||_F^2]_+), \end{aligned} \tag{4.4}$$

where $[\ .\ ]_+ = max(\ .\ , 0)$ and β is a non-negative tuning parameter. In Eq. (4.4), the first term penalizes the large distance between $\mathbf{S}^m$ and its nearest neighbors in $\mathcal{N}(m)$. Specifically, the first term **pulls** $\mathbf{S}^m$ to approximate the average of its nearest neighbors or **pulls** them (*i.e.,* $\mathbf{S}^m$ and its nearest neighbors) together. The second term **pushes** the $\mathbf{S}^m$ against its furthest subjects in $\mathcal{F}(m)$ so that their distance is larger than a fixed margin, *i.e.,* at least a unit "1" in Eq. (4.4), as shown in the third part of Figure 4.1 . In this way, the sparsely connected representation of FCN $\mathbf{S}^m$ is dependent on others $\mathbf{S}^{m'}$ $(m \neq m')$ as well as contains the discriminative ability,

i.e., having a fixed margin to its furthest subjects as well as being close to its nearest neighbors as much as possible, which benefits avoiding the influence of outliers.

Considering Eq. (4.1) and Eq. (4.4), the multi-band fusion model in Eq. (4.1) needs to tune the parameter α, which is time-consuming and needs prior knowledge. In particular, as a personalized framework which trains a fusion model for every test subject, the tuning of parameters is time-consuming. To address this issue, we propose a parameter-free multi-band fusion model as follows:

$$\begin{aligned}\min_{\mathbf{S}^1,...,\mathbf{S}^M} \sum_{m=1}^{M} \sqrt{\sum_{v=1}^{V} ||\mathbf{S}^m - \mathbf{A}^{m,v}||_F^2} + \mathcal{R}(\mathbf{S}^1,...,\mathbf{S}^M) \\ s.t.,\ \forall i, \mathbf{s}_{i,\cdot}^{m^T}\mathbf{1} = 1, s_{i,i}^m = 0, \\ s_{i,j}^m \geq 0\ if\ j \in \mathcal{N}(i),\ otherwise\ 0.\end{aligned} \tag{4.5}$$

Compared Eq. (4.5) with Eq. (4.1), Eq. (4.5) uses a square root operator on the fusion error of each subject to replace the parameter α in Eq. (4.1). Specifically, when we conduct the derivative with respect to $\mathbf{S}^m$, we always get

$$\begin{cases} \min_{\mathbf{S}^m} \lambda^m \dfrac{\partial(\sum_{v=1}^{V} ||\mathbf{S}^m - \mathbf{A}^{m,v}||_F^2)}{\partial \mathbf{S}^m} + \dfrac{\partial(\mathcal{R}(\mathbf{S}^1,...,\mathbf{S}^M))}{\partial \mathbf{S}^m} \\ \quad s.t.,\ \forall i, \mathbf{s}_{i,\cdot}^{m^T}\mathbf{1} = 1, s_{i,i}^m = 0, \\ \quad\quad s_{i,j}^m \geq 0\ if\ j \in \mathcal{N}(i),\ otherwise\ 0. \end{cases} \tag{4.6a}$$

$$\lambda^m = \frac{1}{2\sqrt{\sum_{v=1}^{V} ||\mathbf{S}^m - \mathbf{A}^{m,v}||_F^2}}, \tag{4.6b}$$

where $\boldsymbol{\lambda} = [\lambda^1, ..., \lambda^M]$. Eq. (4.6) is equivalent to Eq. (4.5) for the optimization of $\mathbf{S}^m$ ($m = 1, ..., M$). That is, Eq. (4.5) does not need to tune the parameter by iteratively updating Eq. (4.6a) and Eq. (4.6b) to automatically obtain λ^m during the optimization process. λ^m can be regarded as an implicit parameter, rather than selecting the best one out of a range of values with cross-validation methods used in Eq. (4.1). The value of λ^m is exactly the weight of each subject, indicating the inter-subject variability of each subject. Compared Eq. (4.6) with Eq. (4.1), we have $\lambda^m = 1/\alpha$ for the optimization of the *m*-th FCN $\mathbf{S}^m$. Hence, Eq. (4.1) uses a fixed parameter α while Eq. (4.6) uses a dynamic and data-driven parameter λ^m based on the data distribution.

After optimizing Eq. (4.5), we obtain smooth and sparsely connected representation FCNs $\mathbf{S}^m$ ($m = 1, ..., M$) by fusing multiple fully connected FCNs $\mathbf{A}^{m,v}$ ($v = 1, ..., V$) into a common space. Such a space is spanned by $\mathbf{S}^m$ through its first term as well as shrinks the heterogeneity across subjects through its second term. Moreover, the second term can avoid the influence of outliers by keeping a margin between the nearest neighbor set and the furthest subject set for each subject. Furthermore, we transfer the optimization of Eq. (4.5) to Eq. (4.6) to obtain the trade-off

Algorithm 4.1: The pseudo code of the functional connectivity analysis framework.

Input: $\mathbf{B}^m$ $(m = 1, ..., M)$, $\mathbf{y}$, C;
Output: $\mathbf{S}^m$ $(m = 1, ..., M)$ and the classifier;
Generate $\mathbf{A}^{m,v}(v = 1, ..., V)$ by $\mathbf{B}^m$;
Initialize $\mathbf{S}^m$ as the average of $\mathbf{A}^{m,v}(v = 1, ..., V)$;
repeat
    Update λ^m via Eq. (4.6b);
    Optimize $\mathbf{S}^m$ via Eq. (4.13);
until *Eq. (4.5) converges*;
Generate $\mathbf{X}$ by extracting the upper triangle part of all $\mathbf{S}^m$ $(m = 1, ..., M)$;
Employ ℓ_1-SVM on $\mathbf{X}$ and $\mathbf{y}$ to jointly output top selected brain regions and the classifer;

between two terms by considering the inter-subject variability. Hence, $\mathbf{S}^m$ $(m = 1, ..., M)$ is the new representation of original BOLD signals.

4.3.4 Joint region selection and disease diagnosis

Since the new representation $\mathbf{S}^m$ $(m = 1, ..., M)$ is the matrix representation, it is difficult for using traditional classification methods to conduct disease diagnosis. In this work, we first convert $\mathbf{S}^m$ to a symmetric matrix through a formula $\mathbf{S}^m = (\mathbf{S}^m + (\mathbf{S}^m)^T)/2$, and then follow [203] to transfer the matrix representation to its vector representation, *i.e.,* extracting the upper triangle part of the symmetric matrix $\mathbf{S}^m$ $(m = 1, ..., M)$ to form a row vector $\mathbf{x}_{m,\cdot} \in \mathbb{R}^{1\times n(n-1)/2}$. In this way, we have the data matrix $\mathbf{X} \in \mathbb{R}^{M\times n(n-1)/2}$ and the corresponding label vector $\mathbf{y} \in \{-1, 1\}^{M\times 1}$.

Many previous studies (*e.g.,* [100, 203]) employ a two-stage strategy to conduct disease diagnosis, *i.e.,* feature selection and disease diagnosis. Specifically, in these methods, feature selection is separated from disease diagnosis. Moreover, the goal of feature selection is to preserve the original information as much as possible, rather than to achieve high performance of disease diagnosis. As a result, the best results of feature selection are not good for disease diagnosis. On the contrary, the ℓ_1-SVM (optimized by the public toolbox LIBLINEAR [35]) embeds feature selection by the ℓ_1-norm regularization term on the coefficient matrix with the SVM classifier in the united framework. As a result, the results of feature selection are adjusted based on the classifier updated in the last iteration, while the classifier is also adjusted by the updated results of feature selection. In this way, the results of feature selection contribute to the construction of the classifier. Hence, the ℓ_1-SVM can overcome the drawback of the two-stage strategy in previous methods.

It is noteworthy that the parameter-free multi-band fusion model uses both labelled subjects and unlabelled subjects while the process of joint feature selection and disease diagnosis uses the labelled subjects in the training process.

4.3.5 Optimization, initialization, complexity and convergence

4.3.5.1 Optimization

The proposed objective function in Eq. (4.5) is not convex for all variables, but is convex for any single variable while fixing other variables. Hence, in this chapter, we employ the alternating optimization strategy [27] to iteratively update M FCNs $\mathbf{S}^m$ $(m = 1, ..., M)$ in Eq. (4.5) and list the pseudo of our proposed framework for functional connectivity analysis in Algorithm 4.1.

First, we obtain the expansion formula about $\mathbf{S}^m$ of the first term in Eq. (4.6a) as $V\mathbf{S}^{m^T}\mathbf{S}^m - 2\sum_{v=1}^{V}\mathbf{A}^{m,v^T}\mathbf{S}^m$, and then obtain the expansion result of the second term $\partial(\mathcal{R}(\mathbf{S}^1, ..., \mathbf{S}^M))/\partial\mathbf{S}^m$ about $\mathbf{S}^m$ in Eq. (4.6a) as

$$\begin{cases} k\mathbf{S}^{m^T}\mathbf{S}^m - 2(1+\beta)\sum_{p=1}^{k}\mathbf{S}^{p^T}\mathbf{S}^m + 2\beta\mathbf{D}^m, \ Z^m > 0, \\ k\mathbf{S}^{m^T}\mathbf{S}^m - 2\sum_{p=1}^{k}\mathbf{S}^{p^T}\mathbf{S}^m, \ Z^m \leq 0, \end{cases} \tag{4.7}$$

where $Z^m = 1 + \sum_{p\in\mathcal{N}(m)}||\mathbf{S}^m - \mathbf{S}^p||_F^2 - \sum_{q\in\mathcal{F}(m)}||\mathbf{S}^m - \mathbf{S}^q||_F^2$, $\mathbf{D}^m = \sum_{q=1}^{k}\mathbf{S}^{q^T}\mathbf{S}^m$, and k is the defined number of neighbors for each sample. The optimization of each row $\mathbf{s}_{i,\cdot}^m$ $i = 1, ..., n$, in $\mathbf{S}^m$ is independent on other rows $\mathbf{s}_{i',\cdot}^m$ $(i \neq i')$, so we list the optimization details of $\mathbf{s}_{i,\cdot}^m$ as follows.

$$\min_{\mathbf{s}_{i,\cdot}^{m^T}\mathbf{1}=1, s_{i,i}^m=0, s_{i,j}^m\geq 0} \lambda^m(V\mathbf{s}_{i,\cdot}^{m^T}\mathbf{s}_{i,\cdot}^m - 2\sum_{v=1}^{V}\mathbf{a}_{i,\cdot}^{m,v^T}\mathbf{s}_{i,\cdot}^m) + \sum_{i=1}^{n}\mathcal{R}(\mathbf{s}_{i,\cdot}^1, ..., \mathbf{s}_{i,\cdot}^M), \tag{4.8}$$

where

$$\mathcal{R}(\mathbf{s}_{i,\cdot}^1, ..., \mathbf{s}_{i,\cdot}^M) = \begin{cases} k\mathbf{s}_{i,\cdot}^{m^T}\mathbf{s}_{i,\cdot}^m - 2(1+\beta)\sum_{p=1}^{k}\mathbf{s}_{i,\cdot}^{p^T}\mathbf{s}_{i,\cdot}^m + 2\beta\mathbf{d}_{i,\cdot}^m, \ z_i^m > 0, \\ k\mathbf{s}_{i,\cdot}^{m^T}\mathbf{s}_{i,\cdot}^m - 2\sum_{p=1}^{k}\mathbf{s}_{i,\cdot}^{p^T}\mathbf{s}_{i,\cdot}^m, \ z_i^m \leq 0, \end{cases} \tag{4.9}$$

where $z_i^m = 1 + \sum_{p\in\mathcal{N}(m)}||\mathbf{s}_{i,\cdot}^m - \mathbf{s}_{i,\cdot}^p||_2^2 - \sum_{q\in\mathcal{F}(m)}||\mathbf{s}_{i,\cdot}^m - \mathbf{s}_{i,\cdot}^q||_2^2$ and $\mathbf{d}_{i,\cdot}^m = \sum_{q=1}^{k}\mathbf{s}_{i,\cdot}^{q^T}\mathbf{s}_{i,\cdot}^m$. After finishing mathematical transformation, we have

$$\min_{\mathbf{s}_{i,\cdot}^{m^T}\mathbf{1}=1, s_{i,i}^m=0, s_{i,j}^m\geq 0} ||\mathbf{s}_{i,\cdot}^m - \mathbf{f}_{i,\cdot}^m||_2^2, \tag{4.10}$$

where

$$\mathbf{f}_{i,\cdot}^m = \begin{cases} \frac{\sum_{v=1}^{V}\mathbf{a}_{i,\cdot}^{m,v^T} + (1+\beta)\sum_{p=1}^{k}\mathbf{s}_{i,\cdot}^{p^T} - \beta\sum_{q=1}^{k}\mathbf{s}_{i,\cdot}^{q^T}}{\lambda^m V + k}, \ z_i^m > 0, \\ \frac{\sum_{v=1}^{V}\mathbf{a}_{i,\cdot}^{m,v^T} + \sum_{p=1}^{k}\mathbf{s}_{i,\cdot}^{p^T}}{\lambda^m V + k}, \ z_i^m \leq 0, \end{cases} \tag{4.11}$$

Table 4.1: Demographic information for three data sets. FTD: Fronto-Temporal Dementia; OCD: Obsessive-Compulsive Disorder; ADNI: Alzheimer's Disease Neuroimaging Initiative; AD: Alzheimer's Disease; HC: Healthy Controls.

Data sites	FTD	OCD	ADNI
Age range (years)	65-88	18-50	57-82
Subjects	95 FTD; 86 HC	62 OCD; 20 HC	59 AD; 48 HC
Time points (seconds)	230	230	120
Voxel size (mm^3)	3×3×3	3×3×3	3×3×3
TR(ms)	2000	2000	2000
Scanner	Multi-site	Philips	Multi-site
Field strength	1.5/3T	3T	1.5/3T

where $\mathbf{f}_{i,\cdot}^m \in \mathbb{R}^{n\times 1}$ is a vector. The Lagrangian function with respect to $\mathbf{s}_{i,\cdot}^m$ is

$$\mathcal{L}(\mathbf{s}_{i,\cdot}^m, \sigma, \boldsymbol{\tau}) = ||\mathbf{s}_{i,\cdot}^m - \mathbf{f}_{i,\cdot}^m||_2^2 - \sigma(\mathbf{s}_{i,\cdot}^{m^T}\mathbf{1} - 1) - \boldsymbol{\tau}^T\mathbf{s}_{i,\cdot}^m, \tag{4.12}$$

where $\sigma \in \mathbb{R}$ is the Lagrange multiplier and $\boldsymbol{\tau} \in \mathbb{R}_+^n$ is a non-negative vector. Based on the complementary slackness of the Karush-Kuhn-Tucker (KKT) conditions [11], we have the closed-form solution of $s_{i,j}^m$ is

$$s_{i,j}^m = (f_{i,j}^m + \sigma)_+, \tag{4.13}$$

where $f_{i,j}^m$ is the *j*-th element of $\mathbf{f}_{i,\cdot}^m$. The value of the Lagrange multiplier σ can be obtained by Lemma 4 from [32].

Lemma 4. *By denoting $\mathbf{s}_{i,\cdot}^{m*}$ the optimal solution in Eq. (4.13), letting r and u be two indices, and $f_{i,r}^m > f_{i,u}^m$, only if $s_{i,r}^{m*} = 0$, then $\mathbf{s}_{i,u}^{m*}$ must be equal to zero.*

Based on Lemma 4, we can find some integers $\boldsymbol{I} = [\rho], 1 \leq \rho \leq n$ to meet the non-zero components of the sorted optimal solutions, *i.e.,*

$$\sigma = \frac{1}{\rho}(\sum_{j=1}^{\rho}\mathbf{f}_{i,j}^m - 1). \tag{4.14}$$

As a result, the optimal values in $\mathbf{s}_{i,\cdot}^{m*}$ can be described as $s_{i,j}^{m*} = max\{f_{i,j}^m - \sigma, 0\}$, where the value of the optimal ρ is automatically obtained by Lemma 5 from [32].

Lemma 5. *Let $\boldsymbol{\eta}$ represents the vector after sorting $\mathbf{f}_{i,\cdot}^m$ in a descending order, the number of strictly non-negative elements in $\mathbf{s}_{i,\cdot}^m$ is $\rho = max\left\{\eta_i - \frac{1}{i}(\sum_{l=1}^{i}\eta_l - 1) > 0\right\}, i \in [n]$.*

Based on Lemma 5, the non-zero number in the *i*-th row $\mathbf{s}_{i,\cdot}^m$, *i.e.,* the number of brain regions connected to the *i*-th brain region, is different from the non-zero number in the *j*-th row $\mathbf{s}_{j,\cdot}^m$

($i \neq j$). It is noteworthy that previous sparse methods set the same number of brain regions connected to each brain region. Obviously, our method is more flexible, compared to previous methods in [33, 215, 224].

Furthermore, Eq. (4.5) iteratively updates Eq. (4.6a) and Eq. (4.6b) based on the alternating optimization strategy [27], which has been proved to achieve convergence. Hence, the proposed parameter-free multi-band fusion model converges, while the ℓ_1-SVM also converges.

4.3.5.2 Initialization

In Algorithm 4.1, we initialize $\mathbf{S}^m$ as the average of $\mathbf{A}^{m,v}(v = 1, ..., V)$, which results in that the optimization of Eq. (4.5) converges within tens of iterations. Moreover, the result of Eq. (4.5) is insensitive to the initialization of $\mathbf{S}^m$.

4.3.5.3 Complexity

The generation of both multi-band signals and the fully connected FCNs can be finished offline. The parameter-free multi-band fusion model takes a closed-form solution to optimize $\mathbf{S}^m$ ($m = 1, ..., M$). Its time complexity is $O(Mn^2)$ where *M* and *n*, respectively, represent the number of the subjects and the number of brain regions. That is, the time complexity of our multi-band fusion model is linear to the subject size. Moreover, our model only stores $\mathbf{S}^m$ ($m = 1, ..., M$) in the memory with the space complexity $O(Mn^2)$. The time complexity of ℓ_1-SVM is linear to the subject size, while its space complexity is $O(Mn(n-1)/2)$ [35].

4.3.5.4 Convergence analysis

First, we follow the literature [142, 223] to have the following Lemma:

Lemma 6. *The inequality*

$$\sqrt{u} - \frac{u}{2\sqrt{w}} \leq \sqrt{w} - \frac{w}{2\sqrt{w}} \tag{4.15}$$

holds for non-negative values u *and* w.

Second, Theorem 6 proves the convergence of Algorithm 4.1:

Theorem 1. *The objective function value of Eq. (5) monotonically decreases until Algorithm 4.1 converges.*

Proof. After obtaining the optimal $\mathbf{S}^{m^{(t)}}$ in the t-th iteration, we need to optimize $\mathbf{S}^{m^{(t+1)}}$ in the (t+1)-th iteration by fixing other $\mathbf{S}^{m'^{(t)}}$ where $m' \neq m$, $m = 1, ..., M$.

According to Eq. (13), $s_{i,j}^{m^{(t+1)}}$ has a closed-form solution for all $i, j = 1, ..., n$, so we combine

$\lambda^m = 1/(2\sqrt{\sum_{v=1}^{V} ||\mathbf{S}^{m(t)} - \mathbf{A}^{m,v}||_F^2})$ with Eq. (13) to have:

$$\begin{aligned} &\sum_{m=1}^{M} \frac{\sum_{v=1}^{V} ||\mathbf{S}^{m(t+1)} - \mathbf{A}^{m,v}||_F^2}{2\sqrt{\sum_{v=1}^{V} ||\mathbf{S}^{m(t)} - \mathbf{A}^{m,v}||_F^2}} + \mathcal{R}(\mathbf{S}^{1(t+1)}, ..., \mathbf{S}^{M(t+1)}) \\ &\leq \sum_{m=1}^{M} \frac{\sum_{v=1}^{V} ||\mathbf{S}^{m(t)} - \mathbf{A}^{m,v}||_F^2}{2\sqrt{\sum_{v=1}^{V} ||\mathbf{S}^{m(t)} - \mathbf{A}^{m,v}||_F^2}} + \mathcal{R}(\mathbf{S}^{1(t)}, ..., \mathbf{S}^{M(t)}) \end{aligned} \tag{4.16}$$

Based on Lemma 6, we obtain:

$$\begin{aligned} &\sum_{m=1}^{M} (\sqrt{\sum_{v=1}^{V} ||\mathbf{S}^{m(t+1)} - \mathbf{A}^{m,v}||_F^2} - \frac{\sum_{v=1}^{V} ||\mathbf{S}^{m(t+1)} - \mathbf{A}^{m,v}||_F^2}{2\sqrt{\sum_{v=1}^{V} ||\mathbf{S}^{m(t)} - \mathbf{A}^{m,v}||_F^2}}) \\ &\leq \sum_{m=1}^{M} (\sqrt{\sum_{v=1}^{V} ||\mathbf{S}^{m(t)} - \mathbf{A}^{m,v}||_F^2} - \frac{\sum_{v=1}^{V} ||\mathbf{S}^{m(t)} - \mathbf{A}^{m,v}||_F^2}{2\sqrt{\sum_{v=1}^{V} ||\mathbf{S}^{m(t)} - \mathbf{A}^{m,v}||_F^2}}) \end{aligned} \tag{4.17}$$

Combining Eq. (4.16) with Eq. (4.17), we have:

$$\begin{aligned} &\sum_{m=1}^{M} (\sqrt{\sum_{v=1}^{V} ||\mathbf{S}^{m(t+1)} - \mathbf{A}^{m,v}||_F^2} + \mathcal{R}(\mathbf{S}^{1(t+1)}, ..., \mathbf{S}^{M(t+1)}) \\ &\leq \sum_{m=1}^{M} (\sqrt{\sum_{v=1}^{V} ||\mathbf{S}^{m(t)} - \mathbf{A}^{m,v}||_F^2} + \mathcal{R}(\mathbf{S}^{1(t)}, ..., \mathbf{S}^{M(t)}) \end{aligned} \tag{4.18}$$

Hence, Eq. (4.18) demonstrates Algorithm 4.1 decreases the objective function value of Eq. (4.5) for every iteration until it converges. Therefore, the proof of Theorem 6 is completed.

4.4 Experiments

We experimentally evaluate our method, compared to four comparison methods, on three real neuro-disease data sets with fMRI data, in terms of binary classification performance.

4.4.1 Data sets

The data set fronto-temporal dementia (FTD) contains 95 FTD subjects and 86 age-matched healthy control (HC) subjects, from the recent NIFD database[1], managed by the frontotemporal lobar degeneration neuroimaging initiative. The data set obsessive-compulsive disorder (OCD) from Guangzhou Psychiatric Hospital [31] has 20 HC subjects and 62 OCD subjects. The data set Alzheimer's Disease Neuroimaging Initiative (ADNI)[2] includes 59 Alzheimer's disease (AD) subjects and 48 HC subjects. The demographic information of all data sets is shown in Table 4.1.

[1] https://cind.ucsf.edu/research/grants/frontotemporal-lobar-degeneration-neuroimaging-initiative-0.
[2] http://adni.loni.usc.edu/.

4.4.1.1 Data set OCD

(i) Imaging data acquisition A 3.0-Tesla MR system (Philips Medical Systems, USA) equip with an eight-channel phased-array head coil is used for data acquisition. Functional data are collected using gradient echo Echo-Planar Imaging (EPI) sequences (time repetition, TR = 2000ms; echo time, TE = 60ms; flip angle = 90°, 33 slices, field of view [FOV] = 240mm × 240mm, matrix = 64 × 64; slice thickness = 4.0mm). For spatial normalization and localization, a high-resolution T1-weighted anatomical image is acquired using a magnetization prepared gradient echo sequence (TR = 8ms, TE = 1.7ms, flip angle = 20°, FOV = 240mm × 240mm, matrix = 256 × 256, slice thickness = 1.0mm). During the scanning, participants are instructed to relax with their eyes closed, and stay awake without moving.

(ii) Functional imaging data preprocessing The data are preprocessed using the Statistical Parametric Mapping toolbox (SPM12)[3] and Data Processing Assistant for resting state fMRI (DPARSFA version 4.4)[4]. Image preprocessing consisted of: 1) slice timing correction; 2) head motion correction; 3) realignment with the corresponding T1-volume; 4) nuisance covariate regression (six head motion parameters, white matte signal and cerebrospinal fluid signal); 5) spatial normalization into the stereotactic space of the Montreal Neurological Institute and resampling at $3 \times 3 \times 3mm^3$; 6) spatial smoothing with a 6-mm full-width half-maximum isotropic Gaussian kernel; 7) band-pass filtering (0.01HZ−0.08HZ); 8) micro-head-motion correction according to framewise displacement (FD) by removing the resting state fMRI volume with FD $>$ 0.5 mm (*i.e.,* nearest neighbor interpolation).

4.4.1.2 Data sets FTD and ADNI

For each resting state fMRI scan, we follow the same data processing pipeline on the data set OCD to correct motion and filter BOLD signals for the data sets FTD and ADNI, where the signal bandpass filtering is used to remove the non-brain signal (*i.e.,* beyond 0.01-0.08HZ).

In our experiments, we avoid unnecessary noise of the BOLD signals as much as possible. First, in the acquisition process of BOLD signals, participants are instructed to relax with closed eyes and stay awake without moving so that the obtained data can be trusted to eliminate the external interference. Second, the bandpass filter is used to keep the BOLD signals in the range of 0.01HZ-0.08HZ, aiming at removing the effects of hardware drift within the ultra-low frequency ($<$ 0.01HZ) and the noises (*i.e.,* Gaussian noise, respiratory, and cardiac) within the high frequency ($>$ 0.08HZ) [50, 82, 185]. Third, the beginning part of the original signals is removed as the subjects may not be stable in the rest state at the beginning of the data acquisition [12, 97]. Specifically, we remove the first 30 time points of the signals in the data sets FTD and OCD, and the first 20 time points of the signals in the data set ADNI. Finally, the length of the signals in the data sets FTD and OCD is 200 and the length of the signals in the data set ADNI is 100. As a result, the obtained signals by the above ways can be relevant for brain activity. In this context, it is reasonable to assume the remaining possible noise fall into the random distribution

[3] https://www.fil.ion.ucl.ac.uk/spm.
[4] http://rfmri.org/dpabi.

that is less likely to present across signal bands in a consistent manner. Thus, all BOLD signals in our experiments are pre-processed into the range from 0.01HZ to 0.08HZ, *i.e.,* the so-called original signals in this chapter. The proposed multi-band decomposition is then performed on these original signals in the following steps.

For all imaging data, we follow the automated anatomical labeling (AAL) template [157] to construct the functional connectivity network for each subject with 90 nodes. The region-to-region correlation is measured by Pearson correlation coefficient.

The multi-band is obtained by the following steps. Specifically, the original BOLD data signals are processed by the Discrete Wavelet Transform (DWT), which turn the original signals into multi-band signals. Moreover, the single level DWT is applied with the Daubechies wavelet so that each original data signal will turn into two signals, *i.e.,* the low frequency signal and the high frequency signal, in this work.

4.4.2 Comparison methods

The baseline ℓ_1-SVM (L1SVM) [35] is employed by the public toolbox LIBLINEAR[5] and it uses the least square loss function to conduct the reconstruction error and combines the ℓ_1-norm with the regularization for the final elements selection of feature weight matrix.

High-Order Functional Connectivity (HOFC) [193] learns the correlations across multiple brain regions (*i.e.,* two areas by similarity method or four areas by dynamics method) to conduct the high-order FC from the conventional FC.

Sparse Connectivity Pattern (SCP) [33] finds the common sparsely connected pattern that is a non-negative approximation combination of the fully connected pattern of each subject to all subjects, and the reason is the small part of brain regions can encode the particular activity due to the efficient utilization in the brain.

Simple Graph Convolutional networks (SGC) [169] conducts the simplest graph convolution by removing the non-linear activation (*i.e.,* ReLU [120]) for each graph convolutional layer, and only applies the softmax function for the final classifier construction.

L1SVM is the baseline method, and both HOFC and SCP are the popular methods in neuro-disease diagnosis, and SGC is the deep learning method. L1SVM and SGC extract the vector representation from full FCNs. Other methods (*e.g.,* HOFC, SCP, and our method) design different models to transfer full FCNs to sparse FCNs, and then extract the vector representation from sparse FCNs. Moreover, all methods can be directly applied for supervised learning, while the methods (*e.g.,* SGC and our method) can be used for personalized classification.

[5] https://www.csie.ntu.edu.tw/~cjlin/liblinear/.

Table 4.2: Classification performance (%) of all methods with four evaluation metrics.

Methods	FTD				OCD				ADNI			
	ACC	SEN	SPE	AUC	ACC	SEN	SPE	AUC	ACC	SEN	SPE	AUC
L1SVM	67.58	68.06	63.76	60.09	77.38	72.00	76.57	75.11	74.36	73.80	76.48	78.95
HOFC	78.22	74.36	81.86	60.47	85.35	82.50	86.08	79.54	78.67	79.22	76.99	82.63
SCP	82.67	81.36	77.13	80.98	85.59	81.83	87.80	86.93	85.38	84.97	79.26	87.60
SGC	84.45	87.42	**85.59**	**86.33**	90.08	86.30	90.92	87.28	89.97	86.48	88.37	90.03
Proposed	**86.48**	**87.50**	83.41	86.03	**91.51**	**92.01**	**91.78**	**91.40**	**90.18**	**88.44**	**89.27**	**91.11**

4.4.3 Setting

In our experiments, we repeat the 10-fold cross-validation scheme 10 times to report the average results as the final result, for all methods. In the model selection, we fix $k = 10$ and $\beta = 0.5$ in Eq. (4.4) because they are insensitive based on our experimental results and the literature [165], and further set $C \in \{2^{-6}, 2^{-5}, ..., 2^{6}\}$ for L1SVM. According to the same testing framework, we set the parameters of the comparison methods by following the literature so that they outputted the best results.

We compare our method with the comparison methods by (1) evaluating the performance of supervised learning; (2) evaluating the performance of personalized classification; (3) evaluating the effectiveness of the sparse FCNs outputted by our method; and (4) evaluating the effectiveness of the brain regions selected by our method. The evaluation metrics include ACCuracy (ACC), SPEcificity (SPE), SENsitivity (SEN), and Area Under the receiver operating characteristic Curve (AUC).

4.4.4 Result analysis

4.4.4.1 Supervised learning

We report the results of all methods in Table 4.2. In particular, both our method and SGC only use labelled subjects for the training process.

First, our method obtains the best results, followed by SGC, SCP, HOFC, and L1SVM, in terms of four evaluation metrics. For example, our method improves 18.90%, 19.44%, 19.65%, and 25.94%, compared to the worst method L1SVM, in terms of ACC, SPE, SEN, and AUC, on FTD. Moreover, our method on average improves 2.03%, compared to the best comparison method, *i.e.,* SGC, in terms of accuracy. The reason should be that our fusion model takes discriminative ability and multiple frequency band signals into account, while all comparison methods only use a single frequency band signal.

Second, L1SVM generates fully connected FCNs, while HOFC and SCP output sparse FCNs. SGC uses both fully connected FCNs and a sparse graph, while our method first generates multi-band information and then outputs sparse FCNs. As a result, L1SVM obtains the worst performance. This indicates that sparse FCNs are better than fully connected FCNs for medical image

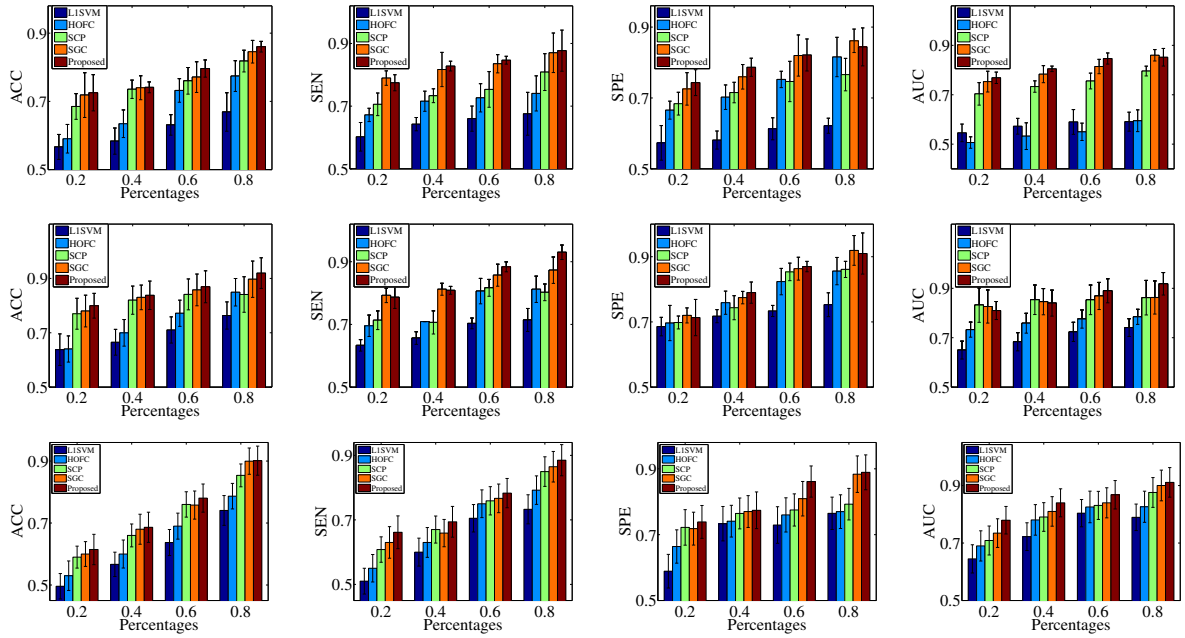


Figure 4.4: Classification results (mean ± standard deviation) of personalized classification on FTD (upper row), OCD (middle row), and ADNI (bottom row).

analysis with fMRI data, as demonstrated in [33, 193].

Third, the methods (*e.g.,* HOFC, SCP, and our method) use different models to convert fully connected FCNs to sparsely connected ones, but our method outperforms the other two. This demonstrates that our multi-band fusion model is the most effective one, compared to either HOFC or SCP. It is noteworthy that SGC outperforms our method in some cases. For example, SGC outperforms our method on the data set FTD, in terms of SPE and AUC. The possible reason is that fully connected FCNs and the sparse graph provide complementary information to each other. However, the deep learning method SGC cannot directly conduct feature selection, so it lacks interpretability.

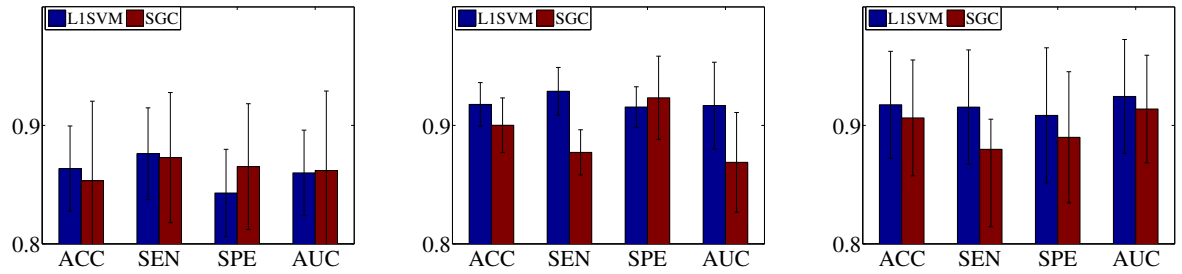
4.4.4.2 Personalized classification

We randomly select different percentages of labelled subjects from the whole data set (*i.e.,* 20%, 40%, 60%, and 80%) as the training set. In this case, L1SVM, HOFC, and SCP only use the labelled subjects to train the classifiers, while our method and SGC use both labelled training subjects and unlabelled test subjects to train the classifier. We report the results in Figure 4.4.

First, our method obtains the best performance at different settings, followed by SGC, SCP, HOFC, and L1SVM. For example, our method on average improves 3.55%, compared to the best comparison method SGC, in terms of all four evaluation metrics, on three data sets with 80% labelled subjects for the training process. Moreover, the performance of supervised learning

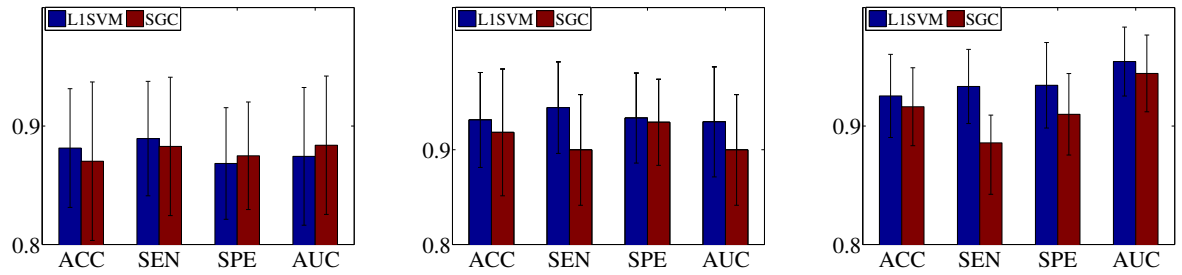
methods (*e.g.,* L1SVM, SCP, and HOFC) in Figure 4.4 is worse than their performance in Table 4.2 since the former use less training information, compared to the latter.

Second, all methods receive worse performance when the percentage of labelled subjects is small. The reason is that inefficient subjects cannot guarantee to construct significant classifiers. However, the improvement of our method over supervised learning methods (*e.g.,* L1SVM, SCP, and HOFC) with small percentages of labelled subjects, *i.e.,* 20%, is larger than its improvement with large percentages of labelled subjects, *e.g.,* 80%, since the former case can use more information than the latter one. The same case can be found in the comparison between SGC and supervised learning methods. This demonstrates the advantages of unlabelled data for the training process again.



(a) FTD (b) OCD (c) ADNI

Figure 4.5: Classification results of the comparison methods using the FCNs outputted by our method.



(a) FTD (b) OCD (c) ADNI

Figure 4.6: Classification results of L1SVM and SGC using the features selected by our method.

4.4.4.3 Fusion effectiveness

All methods first convert the matrix representation (*i.e.,* either the fully connected FCNs or the sparse FCNs) to the vector representation, which is further fed to traditional classifiers for disease diagnosis. The key novelty of our proposed method is the parameter-free multi-band fusion model using multi-band information. Hence, we feed the sparse FCNs produced by our method

(with 80% labelled subjects and 20% unlabelled subjects for the training process) into the methods (*e.g.,* L1SVM and SGC) to verify the effectiveness of our proposed fusion model. We do not apply our outputted FCNs to either HOFC or SCP due to their model limitations. We list the results in Figure 4.5.

The performance of both L1SVM and SGC in Figure 4.5 are better than their performance in the case with 80% labelled subjects in Table 4.2. For example, the result of SGC in Figure 4.5 on average improves 0.88%, 3.53%, and 1.48%, compared to the results in Table 4.2, in terms of accuracy on data sets FTD, OCD, and ADNI. The result of L1SVM in Figure 4.5 on average improves 21.19%, 16.70%, and 18.33%, respectively, compared to the results in Table 4.2, in terms of all four evaluation metrics on FTD, OCD, and ADNI. The reasons are (1) Figure 4.5 used more information (*i.e.,* 20% unlabelled subjects), compared to Table 4.2 for L1SVM; and (2) Figure 4.5 used sparse FCNs, while Table 4.2 used fully connected FCN. Furthermore, the performance of L1SVM in Figure 4.5 is very similar to the performance of our method with 80% labelled data in Table 4.2. The reason is that L1SVM using sparse FCNs produced by our method is exactly our proposed functional connectivity analysis framework.

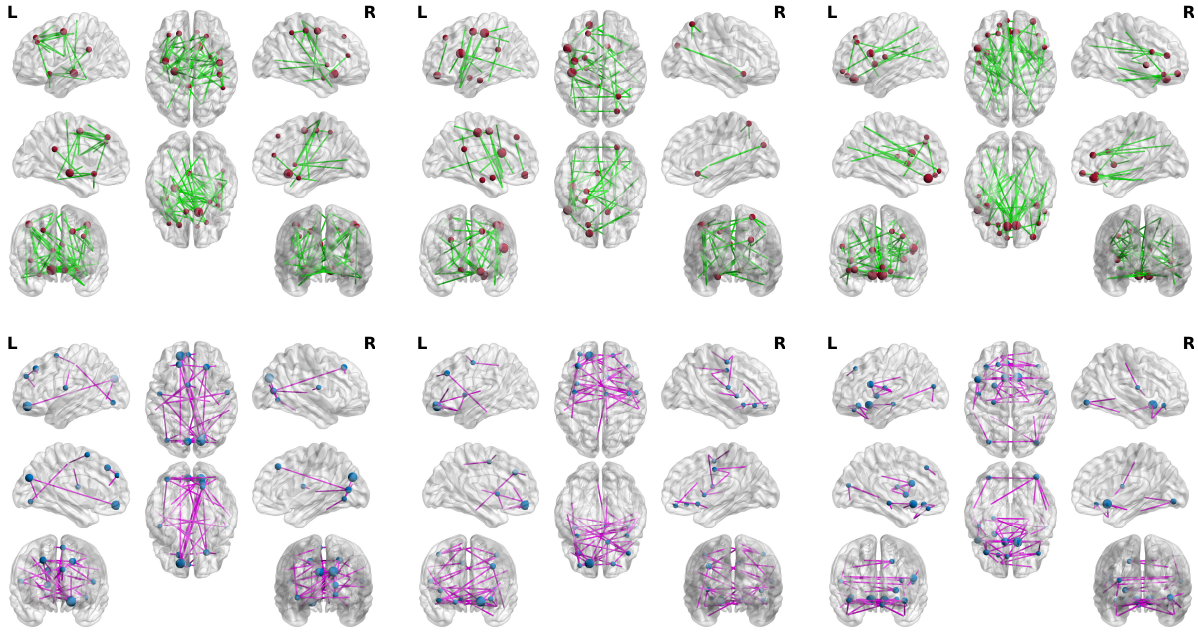


Figure 4.7: Visualization of top selected brain regions selected and the connected regions by L1SVM (top row) and our method (bottom row) on FTD (left column), OCD (middle column), and ADNI (right column).

4.4.4.4 Feature selection effectiveness

In this section, we design two kinds of experiments to investigate the effectiveness of the features selected by our method.

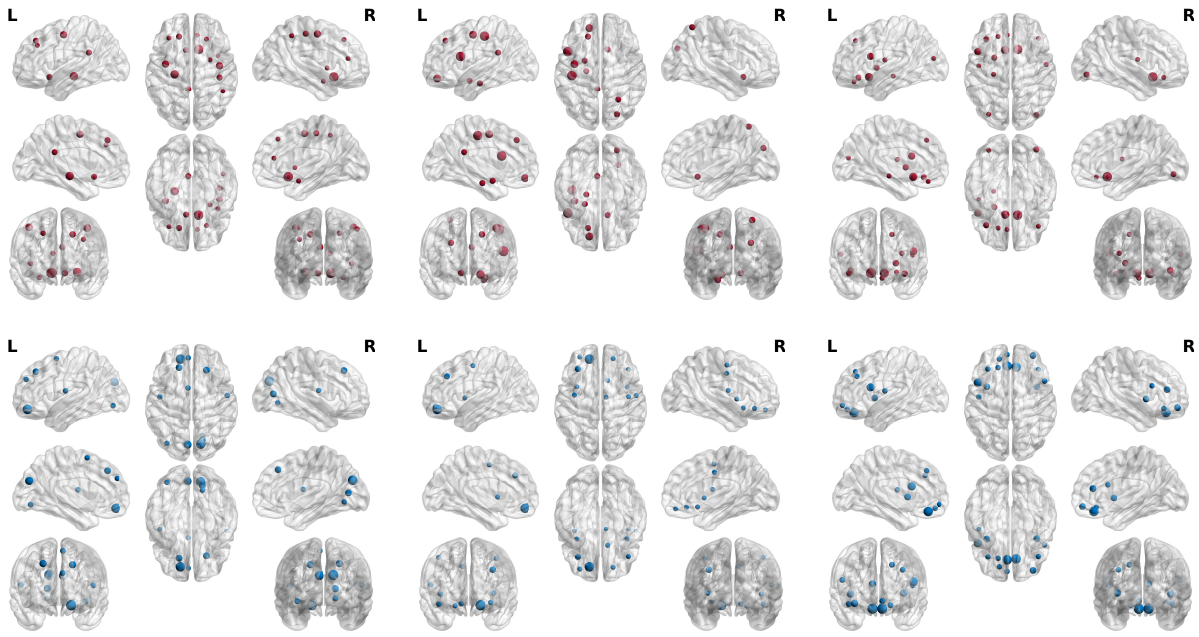


Figure 4.8: Visualization of top selected brain regions selected by L1SVM (top row) and our method (bottom row) on FTD (left column), OCD (middle column), and ADNI (right column).

In our experiments, the 10-fold cross-validation scheme is utilized to evaluate the hyper-parameter C of the ℓ_1-SVM method by setting the values of C as a list $\{2^{-6}, 2^{-5}, ..., 2^{6}\}$. Each fold report the best value of the hyper-parameter C and the corresponding selected features by the ℓ_1-SVM, which result in 10 sets of selected features after the 10-fold cross-validation scheme. These selected features are the most important features in each fold. We repeat these 10-fold cross-validation scheme 10 times with different random seeds and collect all the 100 sets of selected features. The frequencies of the features presented in the 100 sets are then summarized. We select the features that appear at least 90 times as the final set of selected features. Since the features input in the ℓ_1-SVM are the coefficients of correlation between two brain regions, each selected feature have two corresponding brain regions. We summarize the frequencies of the presents of these brain regions in the final set of selected features. Finally, the top 10 brain regions are reported and visualized in this chapter. As a result, our method select 980, 593, and 702 out of 4005 nodes, respectively, on the data sets FTD, OCD, and ADNI, while L1SVM selected 1357, 922, and 970 nodes, respectively. In our experiments, we first apply the nodes selected from our method to L1SVM and SGC for disease diagnosis, and then plot top selected brain regions selected by L1SVM and our method. We report the results in Figure 4.6.

The performance of both L1SVM and SGC in Figure 4.6 is better than their performance in Figure 4.5 because the former use a part of the features (*i.e.,* 980, 593, and 702 out of 4005 nodes, for FTD, OCD, and ADNI, respectively) while the latter use all 4005 features. This implies the effectiveness of feature selection in our method. Furthermore, L1SVM use the features selected by our method means that our method conducts the ℓ_1-SVM twice. For example, the second

Table 4.3: The details of different frequency bands on three data sets.

Band number (g)	FTD /OCD /ADNI
2	[0.01HZ, 0.045HZ] [0.045HZ, 0.08HZ]
3	[0.01HZ, 0.0275HZ] [0.0275HZ, 0.045HZ] [0.045HZ, 0.08HZ]
4	[0.01HZ, 0.01875HZ] [0.01875HZ, 0.0275HZ] [0.0275HZ, 0.045HZ] [0.045HZ, 0.08HZ]

feature selection only uses 1357 features, which is selected by the first feature selection from all 4005 features on FTD. This demonstrates that high-dimensional data may degrade the model performance due to the issue of the curse of dimensionality.

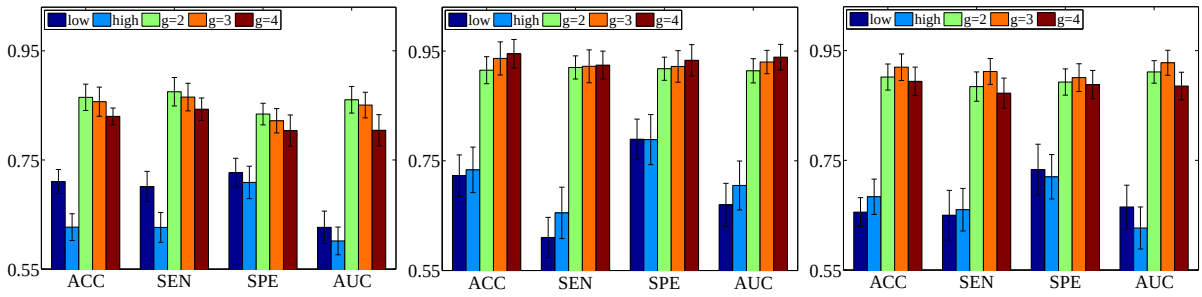
Based on the visualization of the top selected brain regions, many selected regions from both L1SVM and our method have been verified to be related to the neuro-diseases. Moreover, the number of selected brain regions is associated with the neuro-disease with our method is larger than the number selected by L1SVM. This implies that our method is effective for both feature selection and disease diagnosis. Specifically, first, in Figure 4.7, most of the nodes selected by our method occur in the frontal and temporal lobes, which is consistent with the current neurobiological findings on FTD ([28]). However, a large portion of nodes identified by L1SVM located at the occipital lobe and posterior parietal lobe, which are less relevant to FTD ([28]). Second, our method finds the brain regions, such as orbital-frontal cortex, caudate, thalamus, which are included in the cortical-striato-thalamic circuits, and is considered as the theoretical neuro-anatomical network of OCD ([47, 48]). Third, Alzheimer's disease is associated with whole brain atrophy [136], but our method select the brain regions throughout the whole brain while L1SVM only select the frontal regions on the data set ADNI.

4.5 Discussion

In this section, we discuss the effectiveness of the multi-band signals, the variations of our proposed method with different k values, and the convergence analysis of our proposed Algorithm 4.1, using our proposed method (with 80% labelled subjects and 20% unlabelled subjects for the training process) to conduct personalized classification.

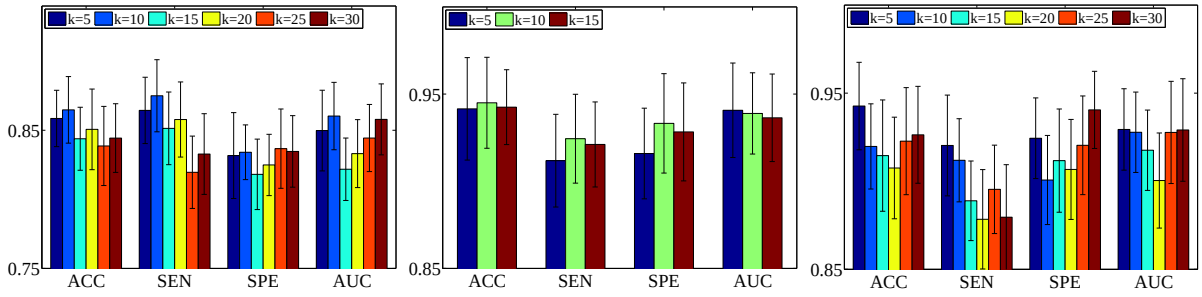
4.5.1 Effectiveness of multi-band signals

We utilize the following steps to obtain multi-band frequency signals. First, the signals' length is padded into the nearest value of 2 powers, guaranteeing the non-destructive wavelet decomposition [49, 206]. Therefore, the signals' length in the data sets FTD and OCD is padded to 256,



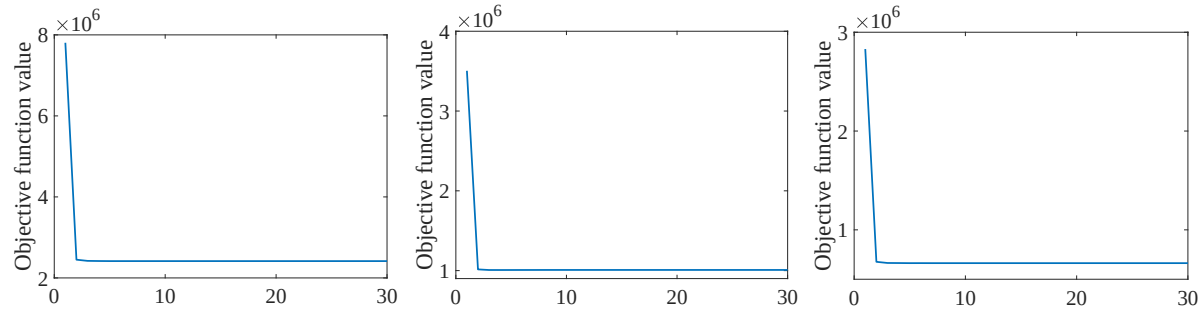
(a) FTD (b) OCD (c) ADNI

Figure 4.9: Classification results (mean $\pm$ standard deviation) of our proposed method with different frequency bands on three data sets, where "low" and "high", respectively, indicate the single frequency band with the range as $[0.01HZ, 0.04HZ]$ and $[0.04HZ, 0.08HZ]$. In particular, "Proposed" is actually the case of $g = 2$.



(a) FTD (b) OCD (c) ADNI

Figure 4.10: Classification results (mean $\pm$ standard deviation) of our proposed method with different values of k on three data sets.



(a) FTD (b) OCD (c) ADNI

Figure 4.11: Convergence analysis of our proposed Algorithm 4.1 at different iterations on three data sets.

and the signals' length in the data set ADNI was padded to 128. To do this, we use the traditional zero-padding technique [117] to reduce the edge effect on the wavelet decomposition. Second,

the Discrete Wavelet Transform (DWT) is performed on the padded signals to decompose them into multi-bands. In our experiments, we follow the traditional Mallat algorithm to decompose the signal into a subband tree [114, 186]. As a result, we control the depth of the subband tree to partition the signals with the frequency range from 0.01HZ to 0.08HZ into four different frequencies, *i.e.,* $[0.01, 0.01875]$, $[0.01875, 0.0275]$, $[0.0275, 0.045]$, and $[0.045, 0.08]$, as shown in Table 4.3. In this chapter, we set experiments to investigate the effectiveness of multi-band signals and report the experimental results of our proposed method in Figure 4.9.

Figure 4.9 indicates the following conclusions. First, our proposed method with multi-band signals (*i.e.,* g = 2/3/4) outperforms our method with the single frequency signals (*i.e.,* "low" and "high"). This verifies that it is reasonable to take into account multi-band signals for the analysis of fMRI data. For example, the classification results with the setting $g = 2$ on the data set FTD on average improves 10.76% and 11.18%, compared to "low" and "high". Second, different data sets require various multi-band signal decomposition. For example, the data set FTD achieved the best classification performance with $g = 2$. The data sets OCD and ADNI, respectively, achieve the best classification performance with $g = 4$ and $g = 3$. Third, with the increase of the decomposition times, the information of the signals in the low frequency bands is gradually diluted and the number of the frequency bands increases. In this manner, the fusion process is more challengeable, compared to the scenarios with fewer decomposition times. The effectiveness of the proposed method is thus affected. That is, there exists the optimal decomposition time. Physiologically, the unsuitable decomposition time (*e.g.,* large values of g in this work) may weaken the ability to disentangle the complex neural activity [134]. For example, [134] find that the signal decomposition with two to four decomposition times can differentiate normal and pathological brain regions.

4.5.2 Effectiveness of k values

Figure 4.10 reports the classification results of our proposed method with different k values on three data sets. Obviously, our proposed method is insensitive to the variations of k value as the difference of the classification results between two different values of k on three data sets is small. For example, our method achieve the best classification result with the k value of 10, 10, and 5, respectively, for the data sets FTD, OCD, and ADNI. These best classification results only averagely increase 2.48%, 0.79%, and 3.10%, respectively, compared to the worst results on the data sets FTD, OCD, and ADNI, in terms of all evaluation metrics.

4.5.3 Convergence analysis

We experimentally analyze the convergence of Algorithm 4.1 in Figure 4.11. Algorithm 4.1 monotonically decreases the objective function values in Eq. (5) until Algorithm 4.1 achieves convergence. Moreover, Algorithm 4.1 only needs around ten iterations to achieve the convergence. Hence, our proposed Algorithm 4.1 is efficient.

4.6 Summary

In this chapter, we proposed a new multi-band fusion framework for personalized disease diagnosis, by two sequential steps, parameter-free multi-band fusion and joint brain region selection & disease diagnosis. In this way, it solved the issue of multi-scale features fusion of graph learning under the semi-supervised learning. Experimental results on three real-world data sets demonstrated the effectiveness of our proposed framework, compared to state-of-the-art methods. Moreover, experimental results also verified the effectiveness of each step in our proposed framework.

This chapter has been published in the CORE rank A* journal, *i.e.,* IEEE Transactions on Medical Imaging [65].

Chapter 5

Multi-scale graph fusion for co-saliency detection

5.1 Introduction

Co-saliency detection focuses on simulating the human visual system to perceive the scene for searching the common and salient prospects from a group of images [188], and has been applied to improve the understanding of the image or video content in various applications such as image retrieval [124], images co-segmentation [156], and object co-localization [74]. In the co-saliency detection task, the semantic category of the common salient objects should be detected from the specific content of the input image group, involving two key steps, *i.e.,* feature extraction extracting discriminative features to reliably distinguish the foregrounds from the backgrounds of each image, and model construction detecting the co-saliency regions from a group of images based on the extracted features.

Feature extraction is focused on extracting either handcrafted features or deep features based on image pixel or superpixel. The popular methods for handcrafted feature extraction include color/texture feature [38], Histogram of Oriented Gradient (HOG) feature [69], GIST descriptors [75], *etc.* Since handcrafted features are usually difficult to capture the appearance changes of both common objects and complex background information [159], deep features have been widely designed to explore the semantic connection of co-saliency objects [156, 188]. For example, [164] propose a group-wise deep co-saliency detection method to consider group-wise features and single image features in the convolutional layers. [129] propose to extract both deep collaborative features and deep high-to-low features to balance the individual intra-image information. [159] employ the VGG-19 framework to extract the high-level group-wise semantic feature and the visual feature for co-saliency detection. Although current feature extraction methods (including handcrafted features and deep features) achieved success in the application of co-saliency detection, extracting single feature is still a challenging task to detect complex variations between co-salient objects and backgrounds. To this end, multi-view feature is extracted

to explore both intra-image and inter-image information for co-saliency detection [77, 196].

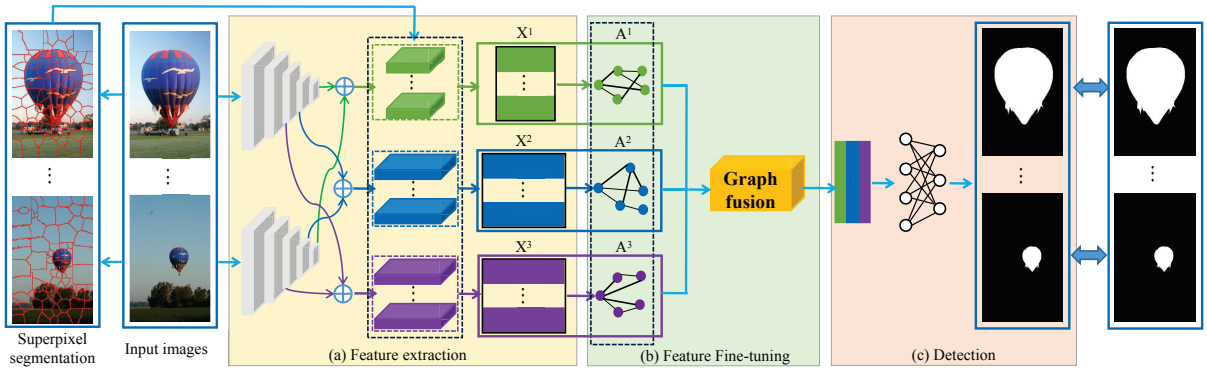


Figure 5.1: The architecture of the proposed framework for co-saliency detection. Specifically, it involves three key steps. (a) Feature extraction extracts three-scale deep features to represent each image; (b) Feature Fine-tuning fine-tunes the multi-scale features to obtain discriminative features by considering their common and complementary information; (c) Detection conducts a binary classification task to distinguish the common salient foregrounds from backgrounds.

Given the image features, both traditional machine learning methods and deep learning methods are designed to detect the co-saliency across a group of images. For example, [191] regard the co-saliency detection task as multi-instance learning where each image and each superpixel region, respectively, are regarded as a bag and an instance, and thus the multi-instance classifier is used to predict the locations of the co-salient objects in the instance level. However, feature extraction and co-saliency detection are two separated processes in many traditional machine learning methods. As a result, the feature can not be adjusted based on the result of co-saliency detection, and thus leading to suboptimal performance of co-saliency detection. To address this issue, deep learning integrates these two processes in a unified framework so that each other can be adaptively adjusted by the other, and thus easily outputting optimal performance of co-saliency detection. For example, fully convolution neural networks are designed to automatically learn high-level semantic features by modeling collaborative correlations among the images [195].

Recently, Graph Convolutional Network (GCN) was designed to utilize both the feature information and the correlation among the images (*i.e.,* the graph) to improve the performance of co-saliency detection [77]. However, previous deep learning methods suffer from some drawbacks to severely limit the detection effectiveness. For example, deep learning methods focus on extracting the self-learnt features from the images without considering the semantic meaning and lacking the interpretability. The convolutional layers and pooling operations in some deep learning methods decrease the size of feature maps to easily result in the loss of boundary details [195].

In this chapter, we propose a novel GCN method to fuse multi-scale features based on the superpixel regions/clusters for co-saliency detection. To do this, our proposed method involves three steps, *i.e.,* feature extraction, feature fusion by the proposed GCN method, and co-saliency detection, shown in Figure 5.1. In the step of feature extraction, we first employ the Simple

Linear Iterative Clustering (SLIC) algorithm [2] to obtain superpixel based regions/clusters including sub-blocks of the background and saliency regions for each image. The motivation is that the superpixel representation may adhere to image boundaries better, compared to pixel representation [188]. We also employ VGG-16 to generate multi-scale features for each image. Furthermore, we convert multi-scale features to represent the image with a vector based on the superpixel clusters. In the step of feature fine-tuning, we design a new graph fusion method to fine-tune the features of each scale by the help of the information of the features from other scales. The goal is to comprehensively explore the intra-image correlation within one image and the inter-region correlation across the images. Finally, the outputted features are concatenated together first and then passes a fully connected layer to conduct the binary classification, *i.e.,* regarding the co-saliency detection task as a classification task.

Compared to previous methods, we list the contributions of our method as follows.

- This chapter first extracts multi-scale features and then designs a new graph fusion method to fine-tune these features. The multi-scale features can detect different sizes of patterns of the images and the fusion method fine-tunes the multi-scale features to extract the complementary information and the common information among the features. It is noteworthy that previous methods [61, 190] extracting handcrafted features are difficult to explore the comprehensive information among the images. Other methods extract the multi-view feature to touch the issue of the handcrafted feature, but leaving the correlation among multiple features alone [77, 107, 196]. Hence, our method is more flexible compared to these methods.
- This chapter proposes a new dynamic GCN method jointly conducting multi-graph fusion, graph learning, and feature learning in a unified work. In the literature, [77] and [196] focused on conducting multi-graph learning on multi-view data by considering the consistency among the graph (*i.e.,* the common information) and ignoring the complementary or private information across multi-scale features.

5.2 Related work

Different from saliency detection in a single image, co-saliency detection captures salient domains of information that have the common meaning for the recognition of a group of related images [102, 155]. Usually, co-saliency detection is influenced by saliency factors in individual images as well as is determined by a consistent saliency map across multiple images. Recently, many methods have been proposed to solve the co-saliency detection problem [38, 159]. Generally, existing work is roughly divided into three groups, including bottom-up methods, fusion methods and learning methods.

Bottom-up method first scores image pixels or super pixels for the co-saliency clues of a group of manually made images, and then combines the corresponding regions in a bottom-up manner. In the early work, manually extracted features (*i.e.,* color, texture and sift descriptor) are widely used [38, 109]. For example, Chang *et al.* conduct the co-saliency detection at the pixel level by

exploring reproducibility between related images [17]. Fu *et al.* [38] propose to mainly explore the global contrast and spatial distribution on a single frame image. However, due to the extracted features relied on the subjective ideas of designers, there are many challenges can be found, such as the clutter of the image background and the appearance square difference of cosine targets between images. Deep learning methods [159, 191] use neural networks to extract differentiated visual features to alleviate the shortcomings of traditional methods in feature representation. For example, Liu *et al.* use the deep learning framework to obtain the internal similarity of the image [178]. Zhang *et al.* use the limited Boltzmann machine to extract the discrimination information and segment the foreground object of the co-significant images [189]. However, these previous methods still ignore the correlation between two saliency regions in one image and the correlation cross images, and neglect plausible saliency proposals that may share high similarity for the same object.

Fusion method employs existing methods or image saliency detection methods to address the above issue [15, 155, 197]. For example, Cao *et al.* use the low rank decomposition to study the correlations of sub-graphs for obtaining adaptive subgraph weights and generating the final co-saliency maps [15]. Huang *et al.* propose integrating multi-scale superpixel to obtain the saliency clue of each image, and employing Gaussian mixture model and binary segmentation to obtain the final co-saliency maps [69]. Tsai *et al.* propose a stacked autoencoder to evaluate the quality of various saliency maps from multiple saliency methods, and conduct a fusion module and a self-trained CNN module for generating a convincing co-saliency maps [155].

Learning method is designed to jointly and automatically learn common patterns and significant clues from image groups, aiming at alleviating the problem that traditional methods in co-saliency detection rely too much on prior knowledge. For example, Zhang *et al.* propose integrating multi-instance learning and self scheduled learning for co-saliency detection [191]. Han *et al.* embed the metric learning regularization into the objective function for co-saliency detection [60]. Wei *et al.* propose to obtain the group interaction information of group images by learning the semantic perception image representation and adaptively learn the group features for collaborative significance detection [164]. Zhang *et al.* use the convolution neural network to calculate the collaborative significance value of the image [190]. Ren *et al.* combine multi-layer features and design an inter image propagation mechanism to generate cooperative co-saliency maps [129]. Tang *et al.* conduct a semantic relation graph and a feature selection way, respectively, to search the inter-saliency relations and important features [150].

5.3 Method

5.3.1 Overview

Denoting $\mathcal{P} = \{\mathbf{P}_n\}_{n=1}^{N}$ as a set of N related images, co-saliency detection is designed to output the map matrix $\mathcal{M} = \{\mathbf{M}^n\}_{n=1}^{N}$, which is used for distinguishing the common salient foregrounds from backgrounds. To this end, our proposed method includes three steps visualized in Figure 5.1.

Given the input image set $\mathcal{P}$, we first employ the SLIC algorithm to conduct superpixel segmentation for obtaining superpixel regions or clusters. Meanwhile, we employ the VGG-16 model [146] to convert each input image to multi-scale images by removing the fully connected layers and the softmax layer of the VGG-16 model. Specifically, we store the images outputted at the third pooling layer, the fourth pooling layer, and the fifth pooling layer. After this, we combine the superpixel regions and the images of each scale to obtain three hierarchical features $\mathcal{X} = \{\mathbf{X}^v\}_{v=1}^3$ as the multi-scale features of $\mathcal{P}$, and thus converting the co-saliency detection task to the classification task based on superpixel clusters.

5.3.2 Feature extraction

Perceptual and semantic visual features are essential for co-saliency detection [187, 188]. A superpixel is usually defined as a set of pixels with common characteristics such as pixel intensity, so superpixel based handcrafted features were shown to carry more semantic information and contain perceptual meaning, compared to either pixel based handcrafted features [43]. However, handcrafted features are not robust to complex visual scenes [129, 205]. On the contrary, deep features can capture the changes within one image or across images to produce robust co-saliency detection models, but lacking semantic meaning. In this chapter, we propose to integrate superpixel segmentation with deep features to generate multi-scale deep features for each image, aiming at producing semantic and discriminative features as well as converting the co-saliency detection task to a classification problem based on the superpixel cluster/region.

Given a set of N related images $\mathcal{P} = \{\mathbf{P}_i\}_{i=1}^N$ ($\mathbf{P}_i \in \mathbb{R}^{224\times224\times3}$, we employ the SLIC algorithm [2] to generate superpixel regions for each image $\mathbf{P}_i$ by clustering pixels based on their color similarity and proximity in the image plane. As a result, we obtain n_i superpixels for each image. For simplicity, we set all n_is ($i = 1, ..., N$) as the same value for a group of images, *i.e., n* and denote $\mathcal{N} = N \times n$. Meanwhile, we input each image $\mathbf{P}_i$ to the pre-trained VGG-16 model to generate three images with different scales, *i.e.,* $\mathbf{P}_i \rightarrow \{\tilde{\mathbf{P}}_i^1, \tilde{\mathbf{P}}_i^2, \tilde{\mathbf{P}}_i^3\}$ where $\tilde{\mathbf{P}}_i^1 \in \mathbb{R}^{56\times56\times256}$, $\tilde{\mathbf{P}}_i^2 \in \mathbb{R}^{28\times28\times512}$, and $\tilde{\mathbf{P}}_i^3 \in \mathbb{R}^{14\times14\times512}$, respectively, denotes the images obtained from the third pooling layer, the fourth pooling layer, and the fifth pooling layer of the VGG-16 model. We then upsampling these images to be the equivalent size, *i.e.,* $\tilde{\mathbf{X}}_i^1 \in \mathbb{R}^{224\times224\times256}$, $\tilde{\mathbf{X}}_i^2 \in \mathbb{R}^{224\times224\times512}$, and $\tilde{\mathbf{X}}_i^3 \in \mathbb{R}^{224\times224\times512}$ where 256 and 512 indicate the filter number, aiming to avoid the loss of boundary details due to the decrease of the feature map size in $\{\tilde{\mathbf{P}}_i^1, \tilde{\mathbf{P}}_i^2, \tilde{\mathbf{P}}_i^3\}$ [43].

For each image $\tilde{\mathbf{X}}_i^j$ ($i = 1, ..., N$ and $j = 1, 2, 3$), we use the result of the superpixel segmentation to partition it into n regions. The representation of each superpixel region is a scalar, which is the average values of the activation maps of all pixels within the same superpixel. Hence, each image is represented by three matrices with different scales of the image size, *e.g.,* $\mathbf{X}_i^1 \in \mathbb{R}^{n\times256}$, $\mathbf{X}_i^2 \in \mathbb{R}^{n\times512}$, and $\mathbf{X}_i^3 \in \mathbb{R}^{n\times512}$, $i = 1, ..., N$. Furthermore, we use $\mathcal{X} = \{\mathbf{X}^1, \mathbf{X}^2, \mathbf{X}^3\}$ to represent the feature matrices of relevant images, where $\mathbf{X}^1 \in \mathbb{R}^{\mathcal{N}\times256}$, $\mathbf{X}^2 \in \mathbb{R}^{\mathcal{N}\times512}$, and $\mathbf{X}^3 \in \mathbb{R}^{\mathcal{N}\times512}$. Finally, the initial graph matrix $\mathbf{A}^v$ ($v = 1, 2, 3$) for $\mathbf{X}^v$ is constructed by the formulation: $\mathbf{A}^v = \mathbf{X}^v\mathbf{X}^{vT} \in \mathbb{R}^{\mathcal{N}\times\mathcal{N}}$ [77].

5.3.3 Feature fine-tuning

In this section, we first review the classical GCN model and then propose our proposed graph fusion method in details.

5.3.3.1 Graph convolutional network

The GCN method aims to learn a latent representation $\mathbf{O}^v = f(\mathbf{X}^v, \mathcal{G}^v; \Theta^v)$ of the original feature matrix $\mathbf{X}^v$ $(v = 1, 2, 3)$ while preserving the graph structure of all data points [87]. Generally, GCN includes one input layer, two hidden layers, and one perceptron layer. Given the input matrix $\mathbf{X}^v \in \mathbb{R}^{\mathcal{N} \times d_v}$ which has $\mathcal{N}$ superpixel regions (or samples) and d_v features for each sample, $\mathbf{A}^v$ denotes the pair-wise correlation between any two samples. Hence, the layer-wise propagation in the k-th hidden layer of GCN is

$$\mathbf{F}^v_{k+1} = \sigma(\tilde{\mathbf{D}}^{v^{-\frac{1}{2}}} \tilde{\mathbf{A}}^v \tilde{\mathbf{D}}^{v^{-\frac{1}{2}}} \mathbf{F}^v_k \mathbf{\Theta}^v_k), \tag{5.1}$$

where $k = (0, 1, ..., K-1)$ and K is the number of layers. $\mathbf{F}^v_0 = \mathbf{X}^v$ is the initial feature matrix, $\mathbf{F}^v_k$ is the output feature map of the k-th layer, $\tilde{\mathbf{A}}^v = \mathbf{A}^v + \mathbf{I}_n$ is the adjacency matrix of the undirected graph, and $\mathbf{I}_n$ is the identity matrix. $\tilde{\mathbf{D}}^v = diag(\tilde{\mathbf{d}}^v_1, ..., \tilde{\mathbf{d}}^v_n)$ is a diagonal matrix with $\tilde{\mathbf{d}}^v_i = \sum_j^n \tilde{\mathbf{A}}^v$ and $\sigma(\cdot)$ is an activation function such as ReLU. The last perception layer is defined as:

$$\mathbf{O}^v = softmax(\tilde{\mathbf{D}}^{v^{-\frac{1}{2}}} \tilde{\mathbf{A}}^v \tilde{\mathbf{D}}^{v^{-\frac{1}{2}}} \mathbf{F}^v_K \mathbf{\Theta}^v_K), \tag{5.2}$$

$\mathbf{O}^v$ is the prediction matrix, $\mathbf{\Theta}^v = (\mathbf{\Theta}^v_0, ..., \mathbf{\Theta}_K{}^v)$ which are trainable parameters and can be learned by minimizing the cross-entropy loss function over the labeled samples.

$$\mathcal{L}_{GCN} : -\sum_{i \in L} \sum_{j}^{c} y_{ij} ln o^v_{ij}, \tag{5.3}$$

where L denotes the set of labelled samples, c is the number of classes, y_{ij} is the ground truth, and o^v_{ij} is the corresponding predictions.

Different from Convolutional Neural Network (CNN) [92] regarding the feature matrix as the input, GCN regards both the feature matrix and the graph as the inputs to generate deep features by preserving the local structure in the graph. As a result, GCN has been demonstrated to outperform CNN in many real applications [87]. Moreover, previous studies (*e.g.,* [24, 78]) showed that the quality of the graph is the key issue for the effectiveness of the GCN method. In the literature, many methods can be used for constructing the graph, *i.e.,* k-nearest neighbor (kNN) graph, ε-nearest neighbor graph, fully connected graph, *etc.* The graph construction by many previous GCN methods is independent of the feature learning process, so that easily resulting in the sub-optimal feature learning. To address this issue, dynamic GCN methods focus on jointly conducting graph learning and feature learning, where the graph can be updated by the optimal features and the features are also adjusted by the updated graph. As a result, the quality of the

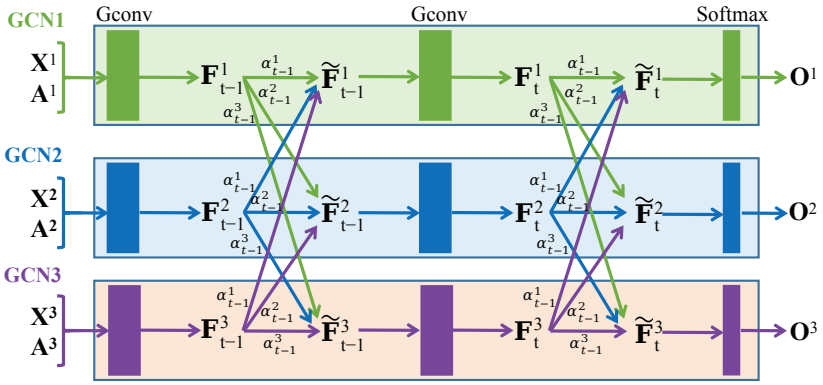


Figure 5.2: The structure of the proposed graph fusion, which conducts feature fine-tuning by exploring the common and complementary information of multi-scale features.

graph can be improved by a data-driven way, and thus the outputted feature is discriminative. To this end, the following objective function of the graph learning is:

$$\begin{aligned} \mathcal{L}_{GL} : \min_{\mathbf{A}^v} & \textstyle\sum_{i,j=1}^{n} \|\mathbf{x}_i^v\mathbf{Q}^v - \mathbf{x}_j^v\mathbf{Q}^v\|_2^2 a_{ij}^v + \|\mathbf{A}^v\|_F^2 \\ & s.t., \ \textstyle\sum_{j=1}^{n} a_{ij}^v = 1, a_{ij}^v > 0, i, j = 1, ..., n, \end{aligned} \tag{5.4}$$

where $\mathbf{Q}^v \in \mathbb{R}^{d_v \times r}$ $(r \leq d_v)$ and a_{ij}^v denote the similarity between $\mathbf{x}_i^v$ and $\mathbf{x}_j^v$. Finally, the dynamic GCN method adds the constraint of the graph learning (*i.e.,* Eq. (5.4)) as the regularization of the GCN model to have:

$$\mathcal{L} = \mathcal{L}_{GCN} + \gamma \mathcal{L}_{GL}, \tag{5.5}$$

where γ is a tuning parameter.

Similar to the literature [77, 101] that approximately optimizes a new variable with less tuning parameters rather than directly optimizing the variable $\mathbf{Q}^v$ in Eq. (5.4) with expensive time cost, this chapter designs to optimize $\mathbf{Q}^v$ by the following objective function:

$$\mathbf{A}^v = \sigma(\mathbf{X}^v\mathbf{Q}^v(\mathbf{X}^v\mathbf{Q})^T), \tag{5.6}$$

where $\sigma(\cdot)$ is the sigmoid activation function and $\mathbf{Q}^v$ is learnable projection matrix.

5.3.3.2 Proposed graph fusion

In this work, we design a new dynamic GCN in Eq. (5.5) and Eq. (5.6) to fine-tune the features of each scale, which was obtained from VGG and superpixel segmentation. Thus we obtain a dynamic GCN model for the features from each scale. However, each GCN model is independently trained from other two. Hence, we propose a fusion method to combine three dynamic GCN models to explore the common information among three models and the complementary information in each model. We list the proposed fusion structure in Figure 5.2.

Specifically, given the feature matrix $\mathbf{X}^v$ and the corresponding graph $\mathbf{A}^v$, the layer-wise propagation in the hidden layer of our proposed GCN method is defined as:

$$\mathbf{F}_t^v = ReLU(\hat{\mathbf{A}}^v \hat{\mathbf{F}}_{(t-1)}^v \mathbf{\Theta}_t^v), \tag{5.7}$$

where $\mathbf{F}_t^v \in \mathbb{R}^{\mathcal{N} \times d_v^t}$ is the new representation of $\hat{\mathbf{F}}_{(t-1)}^v$ in the t-th layer, $\hat{\mathbf{A}}^v = \mathbf{D}^{v^{-\frac{1}{2}}}(\mathbf{A}^v + \mathbf{I}_{\mathcal{N}})\mathbf{D}^{v^{-\frac{1}{2}}}$ is normalized adjacency matrix, and $\mathbf{D}^v$ is the diagonal matrix of $(\mathbf{A}^v + \mathbf{I}_{\mathcal{N}})$. $\mathbf{I}_{\mathcal{N}}$ is an identity matrix and $ReLU(\cdot)$ is an activation function. $\mathbf{\Theta}_t^v$ is a trainable projection matrix for the v-th superpixel feature.

Since we have multi-scale features to describe the same patterns on a group of images. The features with different sizes/scales can capture the common foregrounds with different scales. Moreover, the features of each scale has the complementary information (or private information, *e.g.,* different foregrounds) different from the features from other scales, while all features should have the common information (*i.e.,* the common foregrounds with different scales) as they are assumed to contain the same foregrounds. If the common information is detected, these features will be discriminative for the co-saliency detection. Meanwhile, the difference among the features can also benefit the learning of discriminative features. To this end, we have the definition of $\hat{\mathbf{F}}_{(t-1)}^v$ as follows:

$$\hat{\mathbf{F}}_{(t-1)}^v = \sum_{v=1}^{V} \alpha_{(t-1)}^v \mathbf{F}_{(t-1)}^v, \tag{5.8}$$

where $\alpha_{(t-1)}^v$ indicates the contribution or the weight for $\mathbf{F}_{(t-1)}^v$ to its v-th final features $\hat{\mathbf{F}}_{(t-1)}^v$ in the (t-1)-th layer. Moreover, $\boldsymbol{\alpha}_{(t-1)}^v = [\alpha_{(t-1)}^1, ..., \alpha_{(t-1)}^V]$ is a trainable vector. Specifically, our GCN method outputs $\mathbf{F}_{(t-1)}^v$, which will be combined with all other $\mathbf{F}_{(t-1)}^{v'}$ ($v \neq v'$) to generate the v-th final features $\hat{\mathbf{F}}_{(t-1)}^v$ in the (t-1)-th layer. As a result, the feature learning in each scale have the complementary information (*i.e.,* $\mathbf{F}_{(t-1)}^v$) and the common information from other scales $\hat{\mathbf{F}}_{(t-1)}^{v'}$ ($v \neq v'$).

After the new presentation $\mathbf{F}_t^v$ is obtained, its final output is defined as:

$$\mathbf{O}^v = softmax(\tilde{\mathbf{A}}^v \mathbf{F}_t^v \mathbf{\Theta}_t^v), \tag{5.9}$$

After conducting Eq. (5.9), we obtain three outputs and then concatenate them to have:

$$\mathbf{Z} = FC([\mathbf{O}^1, \mathbf{O}^2, \mathbf{O}^3]), \tag{5.10}$$

where FC denotes the fully connected layer and $\mathbf{Z}$ is the predicted label.

Co-saliency detection is designed to propagate information from intra-superpixel correlations across the relevant images. Hence, we only consider the prediction performance by employing the cross-entropy loss function to obtain:

$$\begin{aligned} \mathcal{L}_{cos} = -\tfrac{1}{\mathcal{N}} \sum_{i=1}^{N} \sum_{j=1}^{n} & \eta^i (\mathbf{z}^i(j) log \mathbf{z}^i(j) \\ & - (1-\eta^i)(1-\mathbf{z}^i(j)) log(1-\mathbf{y}^i(j))), \end{aligned} \tag{5.11}$$

Table 5.1: Results of all methods on three image data sets.

Methods	iCoseg				Cosal2015				MSRC			
	AUC ↑	F_β ↑	S_α ↑	AP ↑	AUC ↑	F_β ↑	S_α ↑	AP ↑	AUC ↑	F_β ↑	S_α ↑	AP ↑
CBCS	0.9315	0.7301	0.6707	0.7958	0.8077	0.5489	0.5439	0.5859	0.8083	0.6563	0.4959	0.6992
ESMG	0.9317	0.7094	0.7436	0.7728	0.7687	0.4803	0.5524	0.5111	0.7875	0.6111	0.5452	0.6112
UMLF	0.9108	0.7262	0.6829	0.7842	0.9011	0.6956	0.6648	0.7398	0.9441	0.8512	0.7982	0.8990
EGNet	0.9598	0.8651	0.8365	0.8751	0.9303	0.7909	0.8206	0.8077	0.8624	0.7714	0.7183	0.7618
MGLCN	0.9671	**0.8912**	0.8355	0.8263	0.9534	0.8845	0.8142	0.8519	0.9415	0.8559	0.8001	0.8427
GCAGC	0.9711	0.8433	0.8334	**0.8813**	0.9676	0.8322	0.8225	0.8765	0.9419	0.8483	0.7904	0.9060
Proposed	**0.9727**	0.8787	**0.8391**	0.8742	**0.9716**	**0.8928**	**0.9341**	**0.8817**	**0.9515**	**0.8565**	**0.8212**	**0.9158**

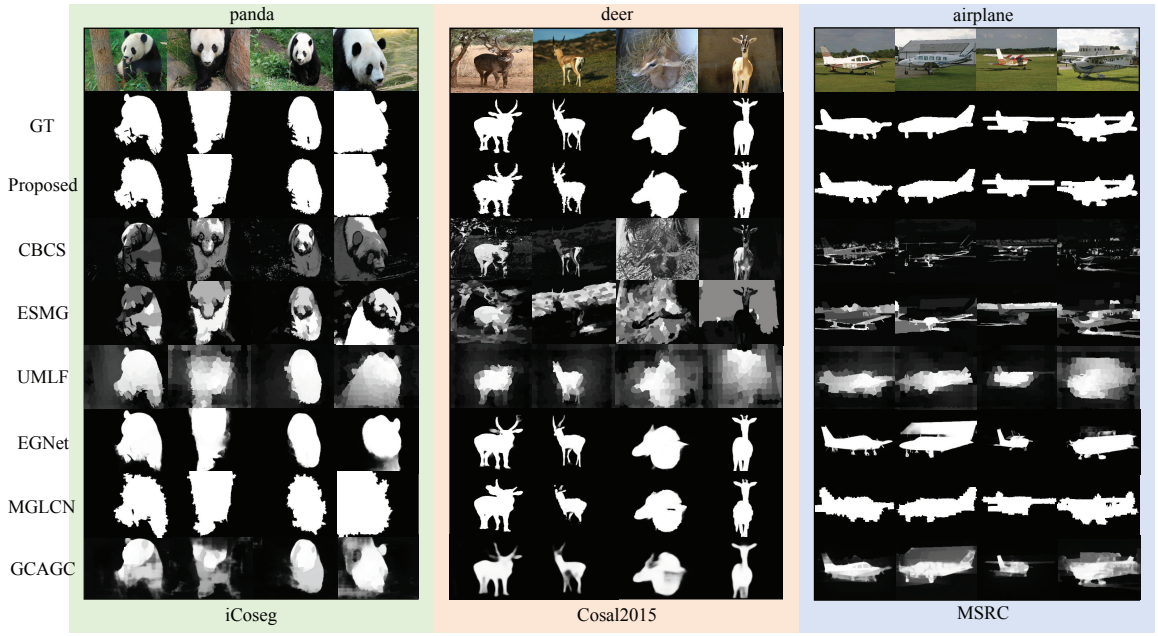


Figure 5.3: Visualization comparisons of all methods on three data sets.

where $\mathbf{y}^i(j)$ and $\mathbf{z}^i(j)$ is the ground truth and predicted result of the *j*-th superpixel of the *i*-th image, respectively. η^i is the ratio of salient superpixel cluster in all superpixel clusters and can be calculated by applying the same superpixel partition for ground truths in advance.

5.4 Experiments

We experimentally evaluate our method, compared to six comparison methods, on three image data sets, *i.e.,* iCoseg, Cosal2015, and MSRC, in terms of six evaluation metrics.

5.4.1 Data sets

The data set iCoseg [6] contains 643 images within 38 different categories. Each image has a manually labeled pixel-wise ground truth for evaluation.

The data set Cosal2015 [190] consists of 2015 images of 50 categories, and each group suffers

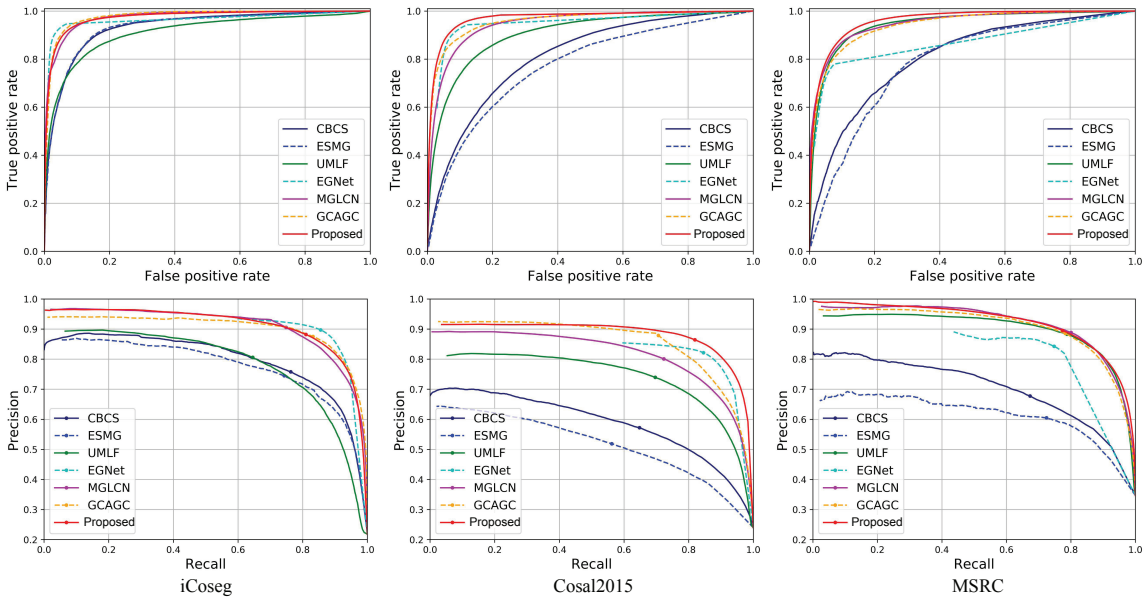


Figure 5.4: ROC and PR curves of all methods on three image data sets.

from various challenging issues such as complex environments, occlusion issues, target appearance variations, and background clutters.

The data set MSRC [168] contains 233 images within 7 categories. The images in the data set are complicated as the common objects are vary unpredictable in color and shape appearance.

5.4.2 Comparison methods

In this chapter, we use six state-of-the-art methods of co-saliency detection to evaluate the effectiveness of our proposed framework, *i.e.,* Cluster-Based Co-Saliency detection (**CBCS**) [38], Efficient Saliency-Model-Guided co-saliency detection (**ESMG**) [103], United Metric Learning-based Framework (**UMLF**) [60], Edge Guidance Network (**EGNet**) [208], Multiple Graph Learning and Convolutional Network (**MGLCN**) [77], and Graph Convolutional network with Attention Graph Clustering (**GCAGC**) [196]. The methods (*e.g.,* CBCS, ESMG, and UMLF) are traditional machine learning methods and the methods (*e.g.,* EGNet, MGLCN, GCAGC, and our method) are deep learning methods.

5.4.3 Setting

In our experiments, we reshape the size of all images to 224×224 and set the number of superpixel regions as 5000. For deep learning methods (*i.e.,* EGNet, MGLCN, GCAGC, and our method), we select the data set MSRAB in [106] to train deep models. In our method, we set the maximal number of epochs as 10000 using the Adam optimizer [86], and set the initial learning rate and the weight decay, respectively, as 0.00005 and 0.005. We set stopping criterion as

no decreasing of the objective function for 100 consecutive epochs in the training process. For fair comparison, we obtained the source codes by online or from the authors. The experimental settings of all comparison methods are followed the corresponding literature to make all of them output their best performance. All experiments are conducted on a server with 4 NVIDIA Quadro P4000 8G.

The evaluation metrics include Precision-Recall (PR) curve, Receiver Operating Characteristic (ROC) curve, Area Under the Curve (AUC) score , F_β score, S_α score, and Average Precision (AP) [34]. Specifically, F_β score is defined as:

$$F_\beta = \frac{(1+\beta^2)Precision \times Recall}{\beta^2 Precision + Recall}, \tag{5.12}$$

where precision and recall are obtained using a self-adaptive threshold $T = \mu + \varepsilon$. μ and ε are the mean and standard deviation values of the saliency map, respectively. We followed [1] to set β^2 as 0.3.

S_α score describes the structural similarity between the ground truths and the corresponding co-saliency maps, and we follow the literature [34] to set all hyper-parameters as 0.5.

5.4.4 Result analysis

We list the results of all methods on three benchmark data sets in Table 5.1, where the bold number stands for the best result in one column. We also report the ROC and PR curves of all methods on all data sets in Figure 5.4.

First, our proposed framework obtain the best performance, followed by GCAGC, MGLCN, EGNet, UMLF, CBCS, and ESMG. For example, our method improved on average by 0.17%, 1.18%, 1.65%, and 0.88%, compared to the best comparison method (*i.e.,* GCAGC), and averagely improved by 4.53%, 9.19%, 8.37%, and 8.63%, compared to the worst comparison method (*i.e.,* ESMG), in terms of AUC, F_β, S_α, and AP, respectively, on three data sets. This indicates the success of our two strategies for co-saliency detection, *i.e.,* generating multi-scale images for every image, and fusing multi-scale features to produce discriminative features. In particular, deep learning methods (*i.e.,* EGNet, MGLCN, GCAGC, and our method) outperform traditional methods (*i.e.,* CBCS, ESMG, and UMLF) as the former methods extract more informative features to describe the salient region than the latter ones. This indicates that deep features are suitable for co-saliency detection.

Second, by comparing with four deep learning methods, EGNet achieve the worst performance as the methods (such as GCAGC, MGLCN, and our method) extract multiple deep features for co-saliency detection. For example, GCAGC improved on average by 1.42%, 1.07%, 0.79%, and 2.44%, respectively, on three data sets, for the evaluation metrics such as AUC, F_β, S_α, and AP, compared to EGNet. This implies that multiple features are reasonable for co-saliency detection.

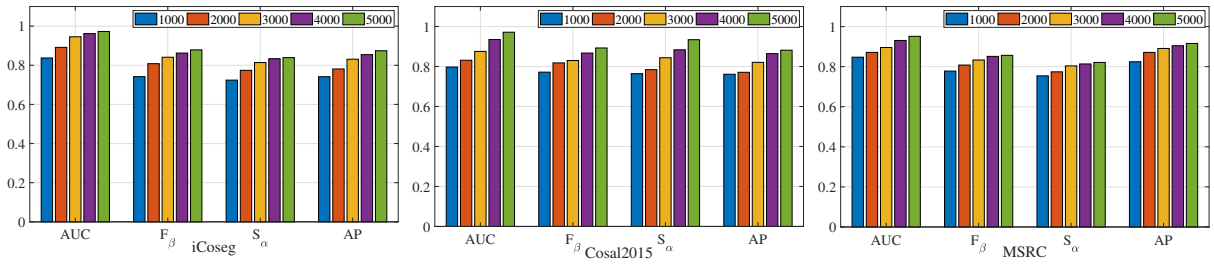


Figure 5.5: Comparison of our framework with different numbers of superpixels on three data sets.

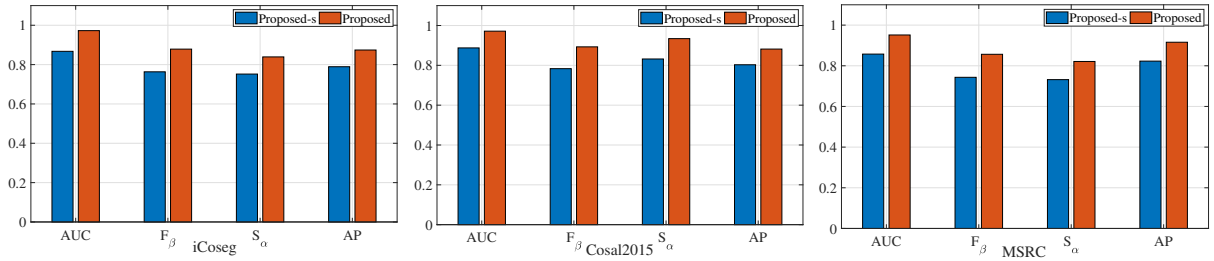


Figure 5.6: Results of our model without/with the process of feature fusion on three data sets.

Figure 5.7: Visual comparisons between the classification task (left) and the regression task (right) using our framework on three images for the used data sets.

5.5 Discussion

In this section, we verify the effectiveness of our model from the following aspects: (1) the number of superpixels for feature extraction; (2) the effectiveness of our fusion method; and (3) regression performance of our method.

5.5.1 Superpixel cluster number

Superpixel segmentation provides a more convenient and compact representation of image representation, compared with the pixel segmentation. Previous studies demonstrated that the performance of co-saliency detection is sensitive the number of superpixel clusters. To this end, we investigate the performance of co-saliency detection with varied number for superpixel clusters,

i.e., 1000, 2000, 3000, 4000, and 5000, and report the corresponding results of our method in Figure 5.5.

Our model obtain the best performance when the superpixel division number is set as 5000. For example, our method with 5000 superpixel clusters improved on average by 10.42%, 7.86%, 6.71%, and 9.13%, respectively, on MSRC data set, in terms of AUC, F_β, S_α, and AP, compared to our method with only 1000 superpixel clusters. Moreover, the more the number of superpixel clusters, the better the performance of the co-saliency detection is. Moreover, the computation cost increases with the increase of the superpixel cluster number. Hence, it is reasonable for us to set superpixel cluster number as 5000 in our experiments by taking into account the trade-off between the effectiveness of co-saliency detection and the computation cost and the effectiveness of our method to detect the common and complementary information from multi-scale features.

5.5.2 Graph fusion effectiveness

In our framework, we fuse the features from multiple scales to explore the complementary information in each scale and the common information among all scales. However, we can also ignore the fusion process, *i.e.,* separately conducting 3 dynamic models and then concatenate 3 outputs to conduct co-saliency detection, Proposed-s for short. We reported the results of both Proposed and Proposed-s in Figure 5.6.

Obviously, Proposed outperformed Proposed-s on all data sets in terms of different evaluation metrics. For example, Proposed improved by on average 9.4%, 11.32%, 8.93%, and 9.29%, respectively, compared to Proposed-s, in terms of AUC, F_β, S_α, and AP. This indicates the importance for feature fusion on multi-scale features.

5.5.3 Regression effectiveness

In this chapter, we regard the co-saliency detection task as a binary classification task, and report the visualization of all methods in Figure 5.3. Actually, we can also regard the co-saliency detection task as a regression task, whose visualization can easier detect the edge boundary compared to the classification task. This is because that the regression task assigns the edge boundary with continuous values and the classification task assigns it with binary values. To this end, we report the visualization of our method on the regression task in Figure 5.7.

Compared the regression task to the classification task in terms of the visualization, the edge boundary produced by the regression task is more blur by considering the pixel graph-scale values, compared to the one in the classification task. Hence, the proposed framework can be designed for both the classification task and the regression task.

5.6 Summary

In this chapter, we proposed a new co-saliency detection framework by designing two strategies to generate discriminative features, *i.e.,* multi-scale features to capture the patterns with

different sizes across the images, and feature fusion to extract the common and complementary information among the multi-scale features. Moreover, we embedded these two strategies into our designed dynamic GCN model to jointly conduct feature fusion, graph learning, and feature learning. In this way, it solved the issue of multiple graph fusion of graph learning within deep learning. Experimental results on three benchmark data sets demonstrated that our framework outperformed the state-of-the-art methods of co-saliency detection in terms of several evaluation metrics. Moreover, experimental results also verified the effectiveness of each strategy in our co-saliency detection framework.

This chapter has been published in the CORE rank A* conference, *i.e.,* The AAAI Conference on Artificial Intelligence 2021 [63].

Chapter 6

Conclusion and future work

6.1 Conclusion

Graph learning method is one of the most popular methods in machine learning and computer vision. Based on the analysis of previous graph learning methods, this thesis focused on the issues of graph learning under supervised learning and semi-supervised learning, including the robustness, the interpretability, the graph fusion for traditional machine learning and deep learning.

Chapter 3 proposed a robust SVM classifier with different constraints, *i.e.,* the hinge loss function, the self-paced learning and the $\ell_{2,1}$-norm sparse learning, to obtain important samples and features in the low-dimensional subspace, aiming at solving the issues of robustness and the interpretability of traditional graph learning. As a result, the proposed method outperformed both classical graph learning methods and dynamic graph learning methods, in terms of binary classification tasks and multi-class classification tasks.

Chapter 4 proposed a multi-band fusion framework under semi-supervised learning to conduct joint graph learning and personalized disease diagnosis, aiming at solving the issues of multi-scale feature fusion. As a result, the proposed method achieved superior performance, compared with both traditional graph learning methods and deep graph learning methods, in terms of functional neuroimaging biomarker identification.

Chapter 5 proposed a deep learning framework by considering multi-scale features extraction, feature fusion, graph learning, and feature learning, to obtain homogeneous graph features in the common feature space, aiming at solving the issues of smoothing multiple graph features. As a consequence, the proposed method outperformed the state-of-the-art traditional methods and deep methods for the co-saliency detection task.

6.2 Future work

Although graph learning has been widely applied in real applications and we also solve some issues in previous graph learning methods, there are still spaces to improve previous graph learning methods to meet the requirements of the real applications.

- Feature learning is the key for deep learning over traditional machine learning because deep learning outputs useful features than traditional machine learning methods with the help of the large number of training data. Graph feature learning is designed to conduct feature learning while preserving the structural information of the data, and thus has attracted much attention. However, this thesis focused on graph learning from supervised learning and semi-supervised learning, while graph feature learning focuses on unsupervised learning and self-supervised learning. Hence, in the future work, we plan to focus on the study of graph feature learning.
- The key component for graph feature learning is the local structure preservation which contains the correlations (*i.e.,* social networks) among the samples. Usually, different local structures explore different correlations of the data. Multiple local structures may enrich the information of the data or even improve the effectiveness of graph feature learning if the multiple local structures provide complementary information to each other. In this way, multiplex graph feature learning uses multiple graph to capture multiple complex correlations of the data, and thus has been attracting much attention. Hence, in the future work, we plan to conduct multiplex graph learning.

References

[1] Radhakrishna Achanta, Sheila Hemami, Francisco Estrada, and Sabine Susstrunk. Frequency-tuned salient region detection. In *CVPR*, pages 1597–1604, 2009.

[2] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Süsstrunk. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11):2274–2282, 2012.

[3] Jeremy Adler and Ingela Parmryd. Quantifying colocalization by correlation: the pearson correlation coefficient is superior to the mander's overlap coefficient. *Cytometry Part A*, 77(8):733–742, 2010.

[4] Maruthamuthu Angulakshmi and Gnanapandithan G. Lakshmi Priya. Brain tumour segmentation from mri using superpixels based spectral clustering. *Journal of King Saud University-Computer and Information Sciences*, 32(10):1182–1193, 2020.

[5] James Atwood and Don Towsley. Diffusion-convolutional neural networks. In *NeurIPS*, volume 29, pages 1–9, 2016.

[6] Dhruv Batra, Adarsh Kowdle, Devi Parikh, Jiebo Luo, and Tsuhan Chen. icoseg: Interactive co-segmentation with intelligent scribble guidance. In *CVPR*, pages 3169–3176, 2010.

[7] Mikhail Belkin, Partha Niyogi, and Vikas Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning Research*, 7(11):2399–2434, 2006.

[8] A.I. Belousov, S.A. Verzakov, and J. Von Frese. Applicational aspects of support vector machines. *Journal of Chemometrics: A Journal of the Chemometrics Society*, 16(8-10):482–489, 2002.

[9] Sandra Benítez-Peña, Rafael Blanquero, Emilio Carrizosa, and Pepa Ramírez-Cobo. Cost-sensitive feature selection for support vector machines. *Computers & Operations Research*, 106:169–178, 2019.

[10] Bernhard E Boser, Isabelle M Guyon, and Vladimir N Vapnik. A training algorithm for optimal margin classifiers. In *COLT*, pages 144–152, 1992.

[11] Stephen Boyd. Convex optimization of graph laplacian eigenvalues. In *ICM*, pages 1311–1319, 2006.

[12] Lisa Byrge and Daniel P Kennedy. Identifying and characterizing systematic temporally-lagged bold artifacts. *NeuroImage*, 171:376–392, 2018.

[13] Deng Cai, Chiyuan Zhang, and Xiaofei He. Unsupervised feature selection for multi-cluster data. In *SIGKDD*, pages 333–342, 2010.

[14] Peng Cao, Dazhe Zhao, and Osmar Zaiane. An optimized cost-sensitive svm for imbalanced data learning. In *PAKDD*, pages 280–292, 2013.

[15] Xiaochun Cao, Zhiqiang Tao, Bao Zhang, Huazhu Fu, and Wei Feng. Self-adaptively weighted co-saliency detection via rank constraint. *IEEE Transactions on Image Processing*, 23(9):4175–4186, 2014.

[16] Chih-Chung Chang and Chih-Jen Lin. Libsvm: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3):1–27, 2011.

[17] Kai-Yueh Chang, Tyng-Luh Liu, and Shang-Hong Lai. From co-saliency to co-segmentation: An efficient and fully unsupervised energy minimization model. In *CVPR*, pages 2129–2136, 2011.

[18] Jie Chen, Tengfei Ma, and Cao Xiao. Fastgcn: fast learning with graph convolutional networks via importance sampling. In *ICLR*, pages 1–15, 2018.

[19] Jingyuan E Chen and Gary H Glover. Bold fractional contribution to resting-state functional connectivity above 0.1 hz. *Neuroimage*, 107:207–218, 2015.

[20] Ming Chen, Zhewei Wei, Zengfeng Huang, Bolin Ding, and Yaliang Li. Simple and deep graph convolutional networks. In *ICML*, pages 1725–1735, 2020.

[21] Xiaobo Chen, Han Zhang, Yue Gao, Chong-Yaw Wee, Gang Li, Dinggang Shen, and Alzheimer's Disease Neuroimaging Initiative. High-order resting-state functional connectivity network for mci classification. *Human Brain Mapping*, 37(9):3282–3296, 2016.

[22] Xiaobo Chen, Han Zhang, Seong-Whan Lee, Dinggang Shen, Alzheimer's Disease Neuroimaging Initiative, et al. Hierarchical high-order functional connectivity networks and selective feature fusion for mci classification. *Neuroinformatics*, 15(3):271–284, 2017.

[23] Xiaobo Chen, Han Zhang, Lichi Zhang, Celina Shen, Seong-Whan Lee, and Dinggang Shen. Extraction of dynamic functional connectivity from brain grey matter and white matter for mci classification. *Human Brain Mapping*, 38(10):5019–5034, 2017.

[24] Yu Chen, Lingfei Wu, and Mohammed Zaki. Iterative deep graph learning for graph neural networks: Better and robust node embeddings. In *NeurIPS*, volume 33, pages 19314–19326, 2020.

[25] Koby Crammer and Yoram Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research*, 2(12):265–292, 2001.

[26] Hoa Khanh Dam, Truyen Tran, Trang Pham, Shien Wee Ng, John Grundy, and Aditya Ghose. Automatic feature learning for predicting vulnerable software components. *IEEE Transactions on Software Engineering*, 47(1):67–85, 2018.

[27] Ingrid Daubechies, Ronald DeVore, Massimo Fornasier, and C Sinan Güntürk. Iteratively reweighted least squares minimization for sparse recovery. *Communications on Pure and Applied Mathematics*, 63(1):1–38, 2010.

[28] Willem de Haan, Yolande AL Pijnenburg, Rob LM Strijers, Yolande van der Made, Wiesje M van der Flier, Philip Scheltens, and Cornelis J Stam. Functional neural network analysis in frontotemporal dementia and alzheimer's disease using eeg and graph theory. *BMC neuroscience*, 10(1):101–112, 2009.

[29] Chris Ding, Ding Zhou, Xiaofeng He, and Hongyuan Zha. R 1-pca: rotational invariant l 1-norm principal component analysis for robust subspace factorization. In *ICML*, pages 281–288, 2006.

[30] Yun Ding, Yuanyuan Guo, Yanwen Chong, Shaoming Pan, and Jinpeng Feng. Global consistent graph convolutional network for hyperspectral image classification. *IEEE Transactions on Instrumentation and Measurement*, 70:1–16, 2021.

[31] Chenjie Dong et al. Impairment in the goal-directed corticostriatal learning system as a biomarker for obsessive–compulsive disorder. *Psychological medicine*, pages 1–11, 2019.

[32] John Duchi, Shai Shalev-Shwartz, Yoram Singer, and Tushar Chandra. Efficient projections onto the l 1-ball for learning in high dimensions. In *ICML*, pages 272–279, 2008.

[33] Harini Eavani, Theodore D Satterthwaite, Roman Filipovych, Raquel E Gur, Ruben C Gur, and Christos Davatzikos. Identifying sparse connectivity patterns in the brain using resting-state fmri. *Neuroimage*, 105:286–299, 2015.

[34] Deng-Ping Fan, Ming-Ming Cheng, Yun Liu, Tao Li, and Ali Borji. Structure-measure: A new way to evaluate foreground maps. In *ICCV*, pages 4548–4557, 2017.

[35] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. Liblinear: A library for large linear classification. *Journal of machine learning research*, 9(8):1871–1874, 2008.

[36] Yanbo Fan, Ran He, Jian Liang, and Baogang Hu. Self-paced learning: an implicit regularization perspective. In *AAAI*, pages 1877–1883, 2017.

[37] Peter Fransson. Spontaneous low-frequency bold signal fluctuations: An fmri investigation of the resting-state default mode of brain function hypothesis. *Human brain mapping*, 26(1):15–29, 2005.

[38] Huazhu Fu, Xiaochun Cao, and Zhuowen Tu. Cluster-based co-saliency detection. *IEEE Transactions on Image Processing*, 22(10):3766–3778, 2013.

[39] Saiji Fu, Xiaotong Yu, and Yingjie Tian. Cost sensitive ν-support vector machine with linex loss. *Information Processing & Management*, 59(2):102809, 2022.

[40] Sichao Fu, Weifeng Liu, Weili Guan, Yicong Zhou, Dapeng Tao, and Changsheng Xu. Dynamic graph learning convolutional networks for semi-supervised classification. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 17(1s):1–13, 2021.

[41] Jiangzhang Gan, Jiaye Li, and Yangcai Xie. Robust svm for cost-sensitive learning. *Neural Processing Letters*, pages 1–22, 2021.

[42] Jiangzhang Gan, Guoqiu Wen, Hao Yu, Wei Zheng, and Cong Lei. Supervised feature selection by self-paced learning regression. *Pattern Recognition Letters*, 132:30–37, 2020.

[43] Guangshuai Gao, Wenting Zhao, Qingjie Liu, and Yunhong Wang. Co-saliency detection with co-attention fully convolutional network. *IEEE Transactions on Circuits and Systems for Video Technology*, 31(3):877–889, 2020.

[44] Wei Gao, Hongtu Zhu, Kelly Giovanello, Keith Smith, Dinggang Shen, John Gilmore, and Weili Lin. Frequency dependence of the developing brain functional network: Mri-eeg? In *ISMRM*, volume 17, page 3432, 2009.

[45] Zoubin Ghahramani. Probabilistic machine learning and artificial intelligence. *Nature*, 521(7553):452–459, 2015.

[46] Philip E Gill and Daniel P Robinson. A primal-dual augmented lagrangian. *Computational Optimization and Applications*, 51(1):1–25, 2012.

[47] Claire M Gillan, Annemieke M Apergis-Schoute, Sharon Morein-Zamir, Gonzalo P Urcelay, Akeem Sule, Naomi A Fineberg, Barbara J Sahakian, and Trevor W Robbins. Functional neuroimaging of avoidance habits in obsessive-compulsive disorder. *American Journal of Psychiatry*, 172(3):284–293, 2015.

[48] Claire M Gillan, Martina Papmeyer, Sharon Morein-Zamir, Barbara J Sahakian, Naomi A Fineberg, Trevor W Robbins, and Sanne de Wit. Disruption in the balance between goal-directed behavior and habit learning in obsessive-compulsive disorder. *American Journal of Psychiatry*, 168(7):718–726, 2011.

[49] Damian Gogolewski. Influence of the edge effect on the wavelet analysis process. *Measurement*, 152:107314, 2020.

[50] Suril R Gohel and Bharat B Biswal. Functional integration between brain regions at rest occurs in multiple-frequency bands. *Brain connectivity*, 5(1):23–34, 2015.

[51] Liyu Gong and Qiang Cheng. Exploiting edge features for graph neural networks. In *CVPR*, pages 9211–9219, 2019.

[52] Bin Gu, Xin Quan, Yunhua Gu, Victor S Sheng, and Guansheng Zheng. Chunk incremental learning for cost-sensitive hinge loss support vector machine. *Pattern Recognition*, 83:196–208, 2018.

[53] Suriya Gunasekar, Blake E Woodworth, Srinadh Bhojanapalli, Behnam Neyshabur, and Nati Srebro. Implicit regularization in matrix factorization. In *NeurIPS*, pages 6151–6159, 2017.

[54] David Gunning, Mark Stefik, Jaesik Choi, Timothy Miller, Simone Stumpf, and Guang-Zhong Yang. Xai—explainable artificial intelligence. *Science Robotics*, 4(37):eaay7120, 2019.

[55] Yanrong Guo, Yaozong Gao, and Dinggang Shen. Deformable MR Prostate Segmentation via Deep Feature Learning and Sparse Patch Matching. *IEEE Transactions on Medical Imaging*, 35(4):1077–1089, 2016.

[56] Zhenhua Guo, Lei Zhang, and David Zhang. A completed modeling of local binary pattern operator for texture classification. *IEEE Transactions on Image Processing*, 19(6):1657–1663, 2010.

[57] Anne Hafkemeijer et al. A longitudinal study on resting state functional connectivity in behavioral variant frontotemporal dementia and alzheimer's disease. *Journal of Alzheimer's Disease*, 55(2):521–537, 2017.

[58] Michael N Hallquist, Kai Hwang, and Beatriz Luna. The nuisance of nuisance regression: spectral misspecification in a common approach to resting-state fmri preprocessing reintroduces noise and obscures functional connectivity. *Neuroimage*, 82:208–225, 2013.

[59] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *NeurIPS*, volume 30, pages 1–11, 2017.

[60] Junwei Han, Gong Cheng, Zhenpeng Li, and Dingwen Zhang. A unified metric learning-based framework for co-saliency detection. *IEEE Transactions on Circuits and Systems for Video Technology*, 28(10):2473–2483, 2017.

[61] Junwei Han, Rong Quan, Dingwen Zhang, and Feiping Nie. Robust object co-segmentation using background prior. *IEEE Transactions on Image Processing*, 27(4):1639–1651, 2017.

[62] Jesper Løve Hinrich, Sophia Elizabeth Bardenfleth, Rasmus Erbou Røge, Nathan William Churchill, Kristoffer Hougaard Madsen, and Morten Mørup. Archetypal analysis for modeling multisubject fmri data. *IEEE journal of selected topics in signal processing*, 10(7):1160–1171, 2016.

[63] Rongyao Hu, Zhenyun Deng, and Xiaofeng Zhu. Multi-scale graph fusion for co-saliency detection. In *AAAI*, volume 35, pages 7789–7796, 2021.

[64] Rongyao Hu, Jiangzhang Gan, Xiaofeng Zhu, Tong Liu, and Xiaoshuang Shi. Multi-task multi-modality svm for early covid-19 diagnosis using chest ct data. *Information Processing & Management*, 59(1):102782, 2022.

[65] Rongyao Hu, Ziwen Peng, Xiaofeng Zhu, Jiangzhang Gan, Yonghua Zhu, Junbo Ma, and Guorong Wu. Multi-band brain network analysis for functional neuroimaging biomarker identification. *IEEE Transactions on Medical Imaging*, 40(12):3843–3855, 2021.

[66] Rongyao Hu, Leyuan Zhang, and Jian Wei. Adaptive laplacian support vector machine for semi-supervised learning. *The Computer Journal*, 64(7):1005–1015, 2021.

[67] Rongyao Hu, Xiaofeng Zhu, Yonghua Zhu, and Jiangzhang Gan. Robust svm with adaptive graph learning. *World Wide Web*, 23(3):1945–1968, 2020.

[68] Heng Huang, Xintao Hu, Yu Zhao, Milad Makkie, Qinglin Dong, Shijie Zhao, Lei Guo, and Tianming Liu. Modeling task fmri data via deep convolutional autoencoder. *IEEE Transactions on Medical Imaging*, 37(7):1551–1561, 2017.

[69] Rui Huang, Wei Feng, and Jizhou Sun. Saliency and co-saliency detection by low-rank multiscale fusion. In *ICME*, pages 1–6, 2015.

[70] Wenbing Huang, Tong Zhang, Yu Rong, and Junzhou Huang. Adaptive sampling towards fast graph representation learning. In *NeurIPS*, volume 31, pages 1–10, 2018.

[71] Zongmo Huang, Yazhou Ren, Xiaorong Pu, Lili Pan, Dezhong Yao, and Guoxian Yu. Dual self-paced multi-view clustering. *Neural Networks*, 140:184–192, 2021.

[72] Arya Iranmehr, Hamed Masnadi-Shirazi, and Nuno Vasconcelos. Cost-sensitive support vector machines. *Neurocomputing*, 343:50–64, 2019.

[73] Mario Jarmasz and Ray L. Somorjai. Exploring regions of interest with cluster analysis (eroica) using a spectral peak statistic for selecting and testing the significance of fmri activation time-series. *Artificial Intelligence in Medicine*, 25(1):45–67, 2002.

[74] Koteswar Rao Jerripothula, Jianfei Cai, Jiangbo Lu, and Junsong Yuan. Object co-skeletonization with co-segmentation. In *CVPR*, pages 3881–3889, 2017.

[75] Koteswar Rao Jerripothula, Jianfei Cai, and Junsong Yuan. Cats: Co-saliency activated tracklet selection for video co-localization. In *ECCV*, pages 187–202, 2016.

[76] Pan Ji, Tong Zhang, Hongdong Li, Mathieu Salzmann, and Ian Reid. Deep subspace clustering networks. In *NeurIPS*, volume 30, pages 1–10, 2017.

[77] Bo Jiang, Xingyue Jiang, Ajian Zhou, Jin Tang, and Bin Luo. A unified multiple graph learning and convolutional network model for co-saliency estimation. In *ACMMM*, pages 1375–1382, 2019.

[78] Bo Jiang, Ziyan Zhang, Doudou Lin, Jin Tang, and Bin Luo. Semi-supervised learning with graph learning-convolutional networks. In *CVPR*, pages 11313–11320, 2019.

[79] Biao Jie, Mingxia Liu, and Dinggang Shen. Integration of temporal and spatial properties of dynamic connectivity networks for automatic diagnosis of brain disease. *Medical image analysis*, 47:81–94, 2018.

[80] Wei Jin, Tyler Derr, Yiqi Wang, Yao Ma, Zitao Liu, and Jiliang Tang. Node similarity preserving graph convolutional networks. In *WSDM*, pages 148–156, 2021.

[81] Wei Jin, Yao Ma, Xiaorui Liu, Xianfeng Tang, Suhang Wang, and Jiliang Tang. Graph structure learning for robust graph neural networks. In *SIGKDD*, pages 66–74, 2020.

[82] Daniel Kalthoff, Jörg U Seehafer, Chrystelle Po, Dirk Wiedermann, and Mathias Hoehn. Functional connectivity in the rat at 11.7 t: Impact of physiological noise in resting state fmri. *Neuroimage*, 54(4):2828–2839, 2011.

[83] Zhao Kang, Haiqi Pan, Steven CH Hoi, and Zenglin Xu. Robust graph learning from noisy data. *IEEE Transactions on Cybernetics*, 50(5):1833–1843, 2019.

[84] Zhao Kang, Chong Peng, Qiang Cheng, Xinwang Liu, Xi Peng, Zenglin Xu, and Ling Tian. Structured graph learning for clustering and semi-supervised classification. *Pattern Recognition*, 110:107627, 2021.

[85] Rose Khavari, Saba N Elias, Timothy Boone, and Christof Karmonik. Similarity of functional connectivity patterns in patients with multiple sclerosis who void spontaneously versus patients with voiding dysfunction. *Neurourology and Urodynamics*, 38(1):239–247, 2019.

[86] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[87] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *ICLR*, pages 1–14, 2016.

[88] Johannes Klicpera, Aleksandar Bojchevski, and Stephan Günnemann. Predict then propagate: Graph neural networks meet personalized pagerank. In *ICLR*, pages 1–15, 2018.

[89] Dehan Kong, Baiguo An, Jingwen Zhang, and Hongtu Zhu. L2rm: Low-rank linear regression models for high-dimensional matrix responses. *Journal of the American Statistical Association*, pages 1–47, 2019.

[90] Dehan Kong, Joseph G Ibrahim, Eunjee Lee, and Hongtu Zhu. Flcrm: Functional linear cox regression model. *Biometrics*, 74(1):109–117, 2018.

[91] Xiang-zhen Kong, Zhaoguo Liu, Lijie Huang, Xu Wang, Zetian Yang, Guangfu Zhou, Zonglei Zhen, and Jia Liu. Mapping individual brain networks using statistical similarity in regional morphology from mri. *PloS One*, 10(11):1–24, 2015.

[92] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NeurIPS*, pages 1097–1105, 2012.

[93] Sofia Ira Ktena, Sarah Parisot, Enzo Ferrante, Martin Rajchl, Matthew Lee, Ben Glocker, and Daniel Rueckert. Metric learning with spectral graph convolutions on brain connectivity networks. *NeuroImage*, 169:431–442, 2018.

[94] M Pawan Kumar, Benjamin Packer, and Daphne Koller. Self-paced learning for latent variable models. In *NeurIPS*, pages 1189–1197, 2010.

[95] Raid Lafta, Ji Zhang, Xiaohui Tao, Yan Li, Mohammed Diykh, and Jerry Chun-Wei Lin. A structural graph-coupled advanced machine learning ensemble model for disease risk prediction in a telehealthcare environment. In *Big Data in Engineering Applications*, pages 363–384. 2018.

[96] Daniel D Langleben and Jane Campbell Moriarty. Using brain imaging for lie detection: Where science, law, and policy collide. *Psychology, Public Policy, and Law*, 19(2):222–244, 2013.

[97] Kaisu Lankinen, Jukka Saari, Yevhen Hlushchuk, Pia Tikka, Lauri Parkkonen, Riitta Hari, and Miika Koskinen. Consistency and similarity of meg-and fmri-signal time courses during movie viewing. *NeuroImage*, 173:361–369, 2018.

[98] Cong Lei and Xiaofeng Zhu. Unsupervised feature selection via local structure learning and sparse learning. *Multimedia Tools and Applications*, 77(22):29605–29622, 2018.

[99] Yunwen Lei, Ürün Dogan, Ding-Xuan Zhou, and Marius Kloft. Data-dependent generalization bounds for multi-class classification. *IEEE Transactions on Information Theory*, 65(5):2995–3021, 2019.

[100] Hongming Li, Theodore D Satterthwaite, and Yong Fan. Large-scale sparse functional networks from resting state fmri. *Neuroimage*, 156:1–13, 2017.

[101] Ruoyu Li, Sheng Wang, Feiyun Zhu, and Junzhou Huang. Adaptive graph convolutional neural networks. In *AAAI*, volume 32, pages 3546–3553, 2018.

[102] Tengpeng Li, Kaihua Zhang, Shiwen Shen, Bo Liu, Qingshan Liu, and Zhu Li. Image co-saliency detection and instance co-segmentation using attention graph clustering based graph convolutional network. *IEEE Transactions on Multimedia*, 24:492–505, 2021.

[103] Yijun Li, Keren Fu, Zhi Liu, and Jie Yang. Efficient saliency-model-guided visual co-saliency detection. *IEEE Signal Processing Letters*, 22(5):588–592, 2014.

[104] Xinxin Liao, Mufang Huang, Wu Xing, Xinwei Wu, Weihua Liao, Xiaoyi Wang, Beisha Tang, and Lu Shen. Resting state fmri studies in spg4-linked hereditary spastic paraplegia. *Journal of the Neurological Sciences*, 384:1–6, 2018.

[105] Luyan Liu, Han Zhang, Jinsong Wu, Zhengda Yu, Xiaobo Chen, Islem Rekik, Qian Wang, Junfeng Lu, and Dinggang Shen. Overall survival time prediction for high-grade glioma patients based on large-scale brain functional networks. *Brain Imaging and Behavior*, 13(5):1333–1351, 2019.

[106] Tie Liu, Zejian Yuan, Jian Sun, Jingdong Wang, Nanning Zheng, Xiaoou Tang, and Heung-Yeung Shum. Learning to detect a salient object. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(2):353–367, 2010.

[107] Yi Liu, Jungong Han, Qiang Zhang, and Caifeng Shan. Deep salient object detection with contextual information guidance. *IEEE Transactions on Image Processing*, 29:360–374, 2019.

[108] Zhenqiu Liu, David Elashoff, and Steven Piantadosi. Sparse support vector machines with l0 approximation for ultra-high dimensional omics data. *Artificial Intelligence in Medicine*, 96:134–141, 2019.

[109] Zhi Liu, Wenbin Zou, Lina Li, Liquan Shen, and Olivier Le Meur. Co-saliency detection based on hierarchical segmentation. *IEEE Signal Processing Letters*, 21(1):88–92, 2013.

[110] Mark J. Lowe, Bryan J. Mock, and James A. Sorenson. Functional connectivity in single and multislice echoplanar imaging using resting-state fluctuations. *Neuroimage*, 7(2):119–132, 1998.

[111] Jianglin Lu, Hailing Wang, Jie Zhou, Yudong Chen, Zhihui Lai, and Qinghua Hu. Low-rank adaptive graph embedding for unsupervised feature extraction. *Pattern Recognition*, 113:107758, 2021.

[112] Guixiang Ma, Nesreen K Ahmed, Ted Willke, Dipanjan Sengupta, Michael W Cole, Nicholas B Turk-Browne, and Philip S Yu. Similarity learning with higher-order graph convolutions for brain network analysis. *arXiv preprint arXiv:1811.02662v5*, 2019.

[113] Paul M Macey, Katherine E Macey, Rajesh Kumar, and Ronald M Harper. A method for removal of global effects from fmri time series. *Neuroimage*, 22(1):360–366, 2004.

[114] Vahid Malekian, Abbas Nasiraei-Moghaddam, Amir Akhavan, and Gholam-Ali Hossein-Zadeh. Efficient de-noising of high-resolution fmri using local and sub-band information. *Journal of Neuroscience Methods*, 331:108497, 2020.

[115] Deyu Meng, Qian Zhao, and Lu Jiang. What objective does self-paced learning indeed optimize? *arXiv preprint arXiv:1511.06049*, 2015.

[116] Di Ming, Chris Ding, and Feiping Nie. A probabilistic derivation of lasso and l12-norm feature selections. In *AAAI*, volume 33, pages 4586–4593, 2019.

[117] Lorenzo Montanari, Biswajit Basu, Andrea Spagnoli, and Brian M Broderick. A padding method to reduce edge effects for enhanced damage identification using wavelet analysis. *Mechanical Systems and Signal Processing*, 52:264–277, 2015.

[118] Vasileios Mygdalis, Anastasios Tefas, and Ioannis Pitas. Learning multi-graph regularization for svm classification. In *ICIP*, pages 1608–1612, 2018.

[119] Vasileios Mygdalis, Anastasios Tefas, and Ioannis Pitas. Exploiting multiplex data relationships in support vector machines. *Pattern Recognition*, 85:70–77, 2019.

[120] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *ICML*, pages 807–814, 2010.

[121] Kevin P Nguyen, Cherise Chin Fatt, Alex Treacher, Cooper Mellema, Madhukar H Trivedi, and Albert Montillo. Anatomically informed data augmentation for functional mri with applications to deep learning. In *Medical Imaging 2020: Image Processing*, volume 11313, pages 1–11, 2020.

[122] Feiping Nie, Yizhen Huang, Xiaoqian Wang, and Heng Huang. New primal svm solver with linear computational cost for big data classifications. In *ICML*, volume 32, pages 505–513, 2014.

[123] Christopher C Paige and Michael A Saunders. Lsqr: An algorithm for sparse linear equations and sparse least squares. *ACM Transactions on Mathematical Software*, 8(1):43–71, 1982.

[124] Alex Papushoy and Adrian G Bors. Image retrieval based on query by saliency content. *Digital Signal Processing*, 36:156–173, 2015.

[125] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12(10):2825–2830, 2011.

[126] Hanyang Peng and Yong Fan. A general framework for sparsity regularized feature selection via iteratively reweighted least square minimization. In *AAAI*, pages 2471–2477, 2017.

[127] Liang Peng, Rongyao Hu, Fei Kong, Jiangzhang Gan, Yujie Mo, Xiaoshuang Shi, and Xiaofeng Zhu. Reverse graph learning for graph neural network. *IEEE Transactions on Neural Networks and Learning Systems*, page DOI:10.1109/TNNLS.2022.3161030, 2022.

[128] Guocheng Qian, Abdulellah Abualshour, Guohao Li, Ali Thabet, and Bernard Ghanem. Pu-gcn: Point cloud upsampling using graph convolutional networks. In *CVPR*, pages 11683–11692, 2021.

[129] Jingru Ren, Zhi Liu, Gongyang Li, Xiaofei Zhou, Cong Bai, and Guangling Sun. Co-saliency detection using collaborative feature extraction and high-to-low feature integration. In *ICME*, pages 1–6, 2020.

[130] Yazhou Ren, Peng Zhao, Yongpan Sheng, Dezhong Yao, and Zenglin Xu. Robust softmax regression for multi-class classification with self-paced learning. In *IJCAI*, pages 2641–2647, 2017.

[131] P Reyes et al. Functional connectivity changes in behavioral, semantic, and nonfluent variants of frontotemporal dementia. *Behavioural neurology*, 2018, 2018.

[132] Yu Rong, Wenbing Huang, Tingyang Xu, and Junzhou Huang. Dropedge: Towards deep graph convolutional networks on node classification. In *ICLR*, pages 1–18, 2019.

[133] Dushyant Sahoo, Christos Davatzikos, and Danielle Bassett. Extraction of hierarchical functional connectivity components in human brain using resting-state fmri. *arXiv preprint arXiv:1906.08365*, 2019.

[134] M. Saritha, K. Paul Joseph, and Abraham T Mathew. Classification of mri brain images using combined wavelet entropy based spider web plots and probabilistic neural network. *Pattern Recognition Letters*, 34(16):2151–2156, 2013.

[135] Saman Sarraf and Ghassem Tofighi. Classification of alzheimer's disease using fmri data and deep learning convolutional neural networks. *arXiv preprint arXiv:1603.08631*, 2016.

[136] Jonathan M. Schott, Sebastian J. Crutch, Chris Frost, Elizabeth K. Warrington, Martin N. Rossor, and Nick C. Fox. Neuropsychological correlates of whole brain atrophy in alzheimer's disease. *Neuropsychologia*, 46(6):1732–1737, 2008.

[137] Katja Seeliger, Umut Güçlü, Luca Ambrogioni, Yagmur Güçlütürk, and Marcel AJ van Gerven. Generative adversarial networks for reconstructing natural images from brain activity. *NeuroImage*, 181:775–785, 2018.

[138] Sadia Shakil, Chin-Hui Lee, and Shella Dawn Keilholz. Evaluation of sliding window correlation performance for characterizing dynamic functional connectivity and brain states. *Neuroimage*, 133:111–128, 2016.

[139] Caifeng Shan, Shaogang Gong, and Peter W McOwan. Facial expression recognition based on local binary patterns: A comprehensive study. *Image and Vision Computing*, 27(6):803–816, 2009.

[140] Yuan-Hai Shao, Chun-Na Li, Ming-Zeng Liu, Zhen Wang, and Nai-Yang Deng. Sparse lq-norm least squares support vector machine with feature selection. *Pattern Recognition*, 78:167–181, 2018.

[141] John Shawe-Taylor and Nello Cristianini. Support vector machines. *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*, pages 93–112, 2000.

[142] Heng Tao Shen, Yonghua Zhu, Wei Zheng, and Xiaofeng Zhu. Half-quadratic minimization for unsupervised feature selection on incomplete data. *IEEE Transactions on Neural Networks and Learning Systems*, 32(7):3122–3135, 2020.

[143] Feng Shi, Yong Fan, Songyuan Tang, John H Gilmore, Weili Lin, and Dinggang Shen. Neonatal brain image segmentation in longitudinal mri studies. *Neuroimage*, 49(1):391–400, 2010.

[144] Hai Shu, Zhe Qu, and Hongtu Zhu. D-gcca: Decomposition-based generalized canonical correlation analysis for multiple high-dimensional datasets. *arXiv preprint arXiv:2001.02856*, 2020.

[145] Hai Shu, Xiao Wang, and Hongtu Zhu. D-cca: A decomposition-based canonical correlation analysis for high-dimensional datasets. *Journal of the American Statistical Association*, pages 1–29, 2019.

[146] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[147] Dinesh Singh and C Krishna Mohan. Graph formulation of video activities for abnormal activity recognition. *Pattern Recognition*, 65:265–272, 2017.

[148] Heung-Il Suk, Chong-Yaw Wee, Seong-Whan Lee, and Dinggang Shen. State-space model with deep learning for functional dynamics estimation in resting-state fmri. *NeuroImage*, 129:292–307, 2016.

[149] Fengzhen Tang, Lukáš Adam, and Bailu Si. Group feature selection with multiclass support vector machine. *Neurocomputing*, 317:42–49, 2018.

[150] Lv Tang, Bo Li, Senyun Kuang, Mofei Song, and Shouhong Ding. Re-thinking the relations in co-saliency detection. *IEEE Transactions on Circuits and Systems for Video Technology*, pages 1–14, 2022.

[151] M Tanveer, A Sharma, and Ponnuthurai N Suganthan. Least squares knn-based weighted multiclass twin svm. *Neurocomputing*, 459:454–464, 2021.

[152] Armin W Thomas, Hauke R Heekeren, Klaus-Robert Müller, and Wojciech Samek. Analyzing neuroimaging data through recurrent deep learning models. *Frontiers in neuroscience*, 13:13–21, 2019.

[153] Garth John Thompson, Wen-Ju Pan, and Shella Dawn Keilholz. Different dynamic resting state fmri patterns are linked to different frequencies of neural activity. *Journal of Neurophysiology*, 114(1):114–124, 2015.

[154] Eric J Topol. High-performance medicine: the convergence of human and artificial intelligence. *Nature Medicine*, 25(1):44–56, 2019.

[155] Chung-Chi Tsai, Kuang-Jui Hsu, Yen-Yu Lin, Xiaoning Qian, and Yung-Yu Chuang. Deep co-saliency detection via stacked autoencoder-enabled fusion and self-trained cnns. *IEEE Transactions on Multimedia*, 22(4):1016–1031, 2019.

[156] Chung-Chi Tsai, Weizhi Li, Kuang-Jui Hsu, Xiaoning Qian, and Yen-Yu Lin. Image co-saliency detection and co-segmentation via progressive joint optimization. *IEEE Transactions on Image Processing*, 28(1):56–71, 2018.

[157] Nathalie Tzourio-Mazoyer et al. Automated anatomical labeling of activations in spm using a macroscopic anatomical parcellation of the mni mri single-subject brain. *Neuroimage*, 15(1):273–289, 2002.

[158] Vladimir Vapnik. Pattern recognition using generalized portrait method. *Automation and Remote Control*, 24:774–780, 1963.

[159] Chunyan Wang, Qiaolin Ye, Peng Luo, Ning Ye, and Liyong Fu. Robust capped l1-norm twin support vector machine. *Neural Networks*, 114:47–59, 2019.

[160] Han Wang, Kun Xie, Li Xie, Xiang Li, Meng Li, Cheng Lyu, Hanbo Chen, Yaowu Chen, Xuesong Liu, Joe Tsien, et al. Functional brain connectivity revealed by sparse coding of large-scale local field potential dynamics. *Brain topography*, 32(2):255–270, 2019.

[161] Jue Wang et al. Frequency-specific alterations of local synchronization in idiopathic generalized epilepsy. *Medicine*, 94(32-40), 2015.

[162] Ping Wang, Jizong Peng, Marco Pedersoli, Yuanfeng Zhou, Caiming Zhang, and Christian Desrosiers. Self-paced and self-consistent co-training for semi-supervised image segmentation. *Medical Image Analysis*, 73:102146, 2021.

[163] Zhen Wang, Long Zhang, Rong Wang, Feiping Nie, and Xuelong Li. Semi-supervised learning via bipartite graph construction with adaptive neighbors. *IEEE Transactions on Knowledge and Data Engineering*, page DOI:10.1109/TKDE.2022.3151315, 2022.

[164] Lina Wei, Shanshan Zhao, Omar El Farouk Bourahla, Xi Li, and Fei Wu. Group-wise deep co-saliency detection. In *IJCAI*, pages 3041–3047, 2017.

[165] Kilian Q Weinberger and Lawrence K Saul. Distance metric learning for large margin nearest neighbor classification. *Journal of machine learning research*, 10(2):207–244, 2009.

[166] Gordon Wetzstein, Aydogan Ozcan, Sylvain Gigan, Shanhui Fan, Dirk Englund, Marin Soljačić, Cornelia Denz, David AB Miller, and Demetri Psaltis. Inference in artificial intelligence with deep optics and photonics. *Nature*, 588(7836):39–47, 2020.

[167] Jennifer L Whitwell and Keith A Josephs. Recent advances in the imaging of frontotemporal dementia. *Current neurology and neuroscience reports*, 12(6):715–723, 2012.

[168] John Winn, Antonio Criminisi, and Thomas Minka. Object categorization by learned universal visual dictionary. In *ICCV*, volume 2, pages 1800–1807, 2005.

[169] Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. Simplifying graph convolutional networks. In *ICML*, pages 6861–6871, 2019.

[170] Jiansheng Wu and Zhihua Zhou. Sequence-based prediction of microrna-binding residues in proteins using cost-sensitive laplacian support vector machines. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 10(3):752–759, 2013.

[171] Xiao-Ming Wu, Zhenguo Li, Anthony So, John Wright, and Shih-Fu Chang. Learning with partially absorbing random walks. In *NeurIPS*, volume 25, pages 1–9, 2012.

[172] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 32(1):4–24, 2020.

[173] Liping Xie, Yong Luo, Shun-Feng Su, and Haikun Wei. Graph regularized structured output svm for early expression detection with online extension. *IEEE Transactions on Cybernetics*, pages 1–13, 2021.

[174] Jinglin Xu, Feiping Nie, and Junwei Han. Feature selection via scaling factor integrated multi-class support vector machines. In *IJCAI*, pages 3168–3174, 2017.

[175] Naganand Yadati, Madhav Nimishakavi, Prateek Yadav, Vikram Nitin, Anand Louis, and Partha Talukdar. Hypergcn: A new method for training graph convolutional networks on hypergraphs. 32:1–12, 2019.

[176] Liu Yang, Yan Yan, Yonghao Wang, Xiaochen Hu, Jie Lu, Piu Chan, Tianyi Yan, and Ying Han. Gradual disturbances of the amplitude of low-frequency fluctuations (alff) and fractional alff in alzheimer spectrum. *Frontiers in neuroscience*, 12:975, 2018.

[177] Dongren Yao, Mingxia Liu, Mingliang Wang, Chunfeng Lian, Jie Wei, Li Sun, Jing Sui, and Dinggang Shen. Triplet graph convolutional network for multi-scale analysis of functional connectivity using functional mri. In *GLMI*, pages 70–78, 2019.

[178] Linwei Ye, Zhi Liu, Junhao Li, Wan-Lei Zhao, and Liquan Shen. Co-saliency detection via co-salient object discovery and recovery. *IEEE Signal Processing Letters*, 22(11):2073–2077, 2015.

[179] Renping Yu, Lishan Qiao, Mingming Chen, Seong-Whan Lee, Xuan Fei, and Dinggang Shen. Weighted graph regularized sparse brain network construction for mci identification. *Pattern Recognition*, 90:220–231, 2019.

[180] Renping Yu, Han Zhang, Le An, Xiaobo Chen, Zhihui Wei, and Dinggang Shen. Correlation-weighted sparse group representation for brain network construction in mci classification. In *MICCAI*, pages 37–45, 2016.

[181] Changan Yuan, Zhi Zhong, Cong Lei, Xiaofeng Zhu, and Rongyao Hu. Adaptive reverse graph learning for robust subspace learning. *Information Processing & Management*, 58(6):102733, 2021.

[182] Guo-Xun Yuan, Kai-Wei Chang, Cho-Jui Hsieh, and Chih-Jen Lin. A comparison of optimization methods and software for large-scale l1-regularized linear classification. *Journal of Machine Learning Research*, 11(11):3183–3234, 2010.

[183] Jing Yuan, Xiang Li, Jinhe Zhang, Liao Luo, Qinglin Dong, Jinglei Lv, Yu Zhao, Xi Jiang, Shu Zhang, Wei Zhang, et al. Spatio-temporal modeling of connectome-scale brain network interactions via time-evolving graphs. *Neuroimage*, 180:350–369, 2018.

[184] Lin Yuan, Xue Wei, Hui Shen, Ling-Li Zeng, and Dewen Hu. Multi-center brain imaging classification using a novel 3d cnn approach. *IEEE Access*, 6:49925–49934, 2018.

[185] Yu-Feng Zang, Yong He, Chao-Zhe Zhu, Qing-Jiu Cao, Man-Qiu Sui, Meng Liang, Li-Xia Tian, Tian-Zi Jiang, and Yu-Feng Wang. Altered baseline brain activity in children

with adhd revealed by resting-state functional mri. *Brain and Development*, 29(2):83–91, 2007.

[186] Wei Zeng, Mengqing Li, Chengzhi Yuan, Qinghui Wang, Fenglin Liu, and Ying Wang. Identification of epileptic seizures in eeg signals using time-scale decomposition (itd), discrete wavelet transform (dwt), phase space reconstruction (psr) and neural networks. *Artificial Intelligence Review*, 53(4):3059–3088, 2020.

[187] Zheng-Jun Zha, Chong Wang, Dong Liu, Hongtao Xie, and Yongdong Zhang. Robust deep co-saliency detection with group semantic and pyramid attention. *IEEE Transactions on Neural Networks and Learning Systems*, 31(7):2398–2408, 2020.

[188] Dingwen Zhang, Huazhu Fu, Junwei Han, Ali Borji, and Xuelong Li. A review of co-saliency detection algorithms: Fundamentals, applications, and challenges. *ACM Transactions on Intelligent Systems and Technology*, 9(4):1–31, 2018.

[189] Dingwen Zhang, Junwei Han, Chao Li, and Jingdong Wang. Co-saliency detection via looking deep and wide. In *CVPR*, pages 2994–3002, 2015.

[190] Dingwen Zhang, Junwei Han, Chao Li, Jingdong Wang, and Xuelong Li. Detection of co-salient objects by looking deep and wide. *International Journal of Computer Vision*, 120(2):215–232, 2016.

[191] Dingwen Zhang, Deyu Meng, and Junwei Han. Co-saliency detection via a self-paced multiple-instance learning framework. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(5):865–878, 2016.

[192] Dingwen Zhang, Deyu Meng, Long Zhao, and Junwei Han. Bridging saliency detection to weakly supervised object detection based on self-paced curriculum learning. *arXiv preprint arXiv:1703.01290*, 2017.

[193] Han Zhang, Xiaobo Chen, Yu Zhang, and Dinggang Shen. Test-retest reliability of "high-order" functional connectivity in young healthy adults. *Frontiers in neuroscience*, 11:439–458, 2017.

[194] Ji Zhang, Leonard Tan, and Xiaohui Tao. On relational learning and discovery in social networks: a survey. *International Journal of Machine Learning and Cybernetics*, 10(8):2085–2102, 2019.

[195] Kaihua Zhang, Tengpeng Li, Bo Liu, and Qingshan Liu. Co-saliency detection via mask-guided fully convolutional networks with multi-scale label smoothing. In *CVPR*, pages 3095–3104, 2019.

[196] Kaihua Zhang, Tengpeng Li, Shiwen Shen, Bo Liu, Jin Chen, and Qingshan Liu. Adaptive graph convolutional network with attention graph clustering for co-saliency detection. In *CVPR*, pages 9050–9059, 2020.

[197] Qijian Zhang, Runmin Cong, Junhui Hou, Chongyi Li, and Yao Zhao. Coadnet: Collaborative aggregation-and-distribution networks for co-salient object detection. 33:6959–6970, 2020.

[198] Wei Zhang, Jinglei Lv, Xiang Li, Dajiang Zhu, Xi Jiang, Shu Zhang, Yu Zhao, Lei Guo, Jieping Ye, Dewen Hu, et al. Experimental comparisons of sparse dictionary learning and independent component analysis for brain network inference from fmri data. *IEEE Transactions on Biomedical Engineering*, 66(1):289–299, 2018.

[199] Xiaobo Zhang, Yan Yang, Tianrui Li, Hao Wang, and Ziqing He. Discovering senile dementia from brain mri using ra-densenet. In *PAKDD*, pages 449–460, 2019.

[200] Yangsong Zhang, Erwei Yin, Fali Li, Yu Zhang, Daqing Guo, Dezhong Yao, and Peng Xu. Hierarchical feature fusion framework for frequency recognition in ssvep-based bcis. *Neural Networks*, 119:1–9, 2019.

[201] Yin Zhang and Zhihua Zhou. Cost-sensitive face recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(10):1758–1769, 2009.

[202] Yu Zhang, Han Zhang, Xiaobo Chen, Seong-Whan Lee, and Dinggang Shen. Hybrid high-order functional connectivity networks using resting-state functional mri for mild cognitive impairment diagnosis. *Scientific Reports*, 7(1):6530–6545, 2017.

[203] Yu Zhang, Han Zhang, Xiaobo Chen, Mingxia Liu, Xiaofeng Zhu, Seong-Whan Lee, and Dinggang Shen. Strength and similarity guided group-level brain functional network construction for mci diagnosis. *Pattern Recognition*, 88:421–430, 2019.

[204] Yu Zhang, Han Zhang, Xiaobo Chen, and Dinggang Shen. Constructing multi-frequency high-order functional connectivity network for diagnosis of mild cognitive impairment. In *CNI*, pages 9–16, 2017.

[205] Zhao Zhang, Wenda Jin, Jun Xu, and Ming-Ming Cheng. Gradient-induced co-saliency detection. In *ECCV*, pages 455–472, 2020.

[206] Zhenyou Zhang, Yi Wang, and Kesheng Wang. Fault diagnosis and prognosis using wavelet packet decomposition, fourier transform and artificial neural network. *Journal of Intelligent Manufacturing*, 24(6):1213–1227, 2013.

[207] Ziwei Zhang, Peng Cui, and Wenwu Zhu. Deep learning on graphs: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 34(1):249–270, 2020.

[208] Jia-Xing Zhao, Jiang-Jiang Liu, Deng-Ping Fan, Yang Cao, Jufeng Yang, and Ming-Ming Cheng. Egnet: Edge guidance network for salient object detection. In *ICCV*, pages 8779–8788, 2019.

[209] Wei Zheng, Xiaofeng Zhu, Guoqiu Wen, Yonghua Zhu, Hao Yu, and Jiangzhang Gan. Unsupervised feature selection by self-paced learning regularization. *Pattern Recognition Letters*, 132:4–11, 2020.

[210] Wei Zheng, Xiaofeng Zhu, Yonghua Zhu, Rongyao Hu, and Cong Lei. Dynamic graph learning for spectral feature selection. *Multimedia Tools and Applications*, 77(22):29739–29755, 2018.

[211] Wei Zheng, Xiaofeng Zhu, Yonghua Zhu, and Shichao Zhang. Robust feature selection on incomplete data. In *IJCAI*, pages 3191–3197, 2018.

[212] Yuanjie Zheng, Jingyi Yu, Chandra Kambhamettu, Sarah Englander, Mitchell D Schnall, and Dinggang Shen. De-enhancing the dynamic contrast-enhanced breast mri for robust registration. In *MICCAI*, pages 933–941, 2007.

[213] Dengyong Zhou, Jiayuan Huang, and Bernhard Schölkopf. Learning with hypergraphs: Clustering, classification, and embedding. 19:1–8, 2006.

[214] Shenglong Zhou. Sparse svm for sufficient data reduction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–11, 2021.

[215] Yueying Zhou, Limei Zhang, Shenghua Teng, Lishan Qiao, and Dinggang Shen. Improving sparsity and modularity of high-order functional connectivity networks for mci and asd identification. *Frontiers in neuroscience*, 12:959–970, 2018.

[216] Ji Zhu, Saharon Rosset, Robert Tibshirani, and Trevor J Hastie. 1-norm support vector machines. In *NeurIPS*, pages 49–56, 2004.

[217] Xiaofeng Zhu, Jiangzhang Gan, Guangquan Lu, Jiaye Li, and Shichao Zhang. Spectral clustering via half-quadratic optimization. *World Wide Web*, 23(3):1969–1988, 2020.

[218] Xiaofeng Zhu, Wei He, Yonggang Li, Yang Yang, Shichao Zhang, Rongyao Hu, and Yonghua Zhu. One-step spectral clustering via dynamically learning affinity matrix and subspace. In *AAAI*, volume 31, pages 2963–2969, 2017.

[219] Xiaofeng Zhu, Xuelong Li, Shichao Zhang, Zongben Xu, Litao Yu, and Can Wang. Graph pca hashing for similarity search. *IEEE Transactions on Multimedia*, 19(9):2033–2044, 2017.

[220] Xiaofeng Zhu, Gerard Sanroma, Jilian Zhang, and Brent C Munsell. Deep mining big social data. *World Wide Web*, 21(6):1449–1452, 2018.

[221] Xiaofeng Zhu, Shichao Zhang, Rongyao Hu, Wei He, Cong Lei, and Pengfei Zhu. One-step multi-view spectral clustering. *IEEE Transactions on Knowledge and Data Engineering*, 31(10):2022–2034, 2019.

[222] Xiaofeng Zhu, Shichao Zhang, Yonggang Li, Jilian Zhang, Lifeng Yang, and Yue Fang. Low-rank sparse subspace for spectral clustering. *IEEE Transactions on Knowledge and Data Engineering*, 31(8):1532–1543, 2019.

[223] Xiaofeng Zhu, Shichao Zhang, Yonghua Zhu, Pengfei Zhu, and Yue Gao. Unsupervised spectral feature selection with dynamic hyper-graph learning. *IEEE Transactions on Knowledge and Data Engineering*, page 10.1109/TKDE.2020.3017250, 2020.

[224] Pascal Zille, Vince D Calhoun, Julia M Stephen, Tony W Wilson, and Yu-Ping Wang. Fused estimation of sparse connectivity patterns from rest fmri—application to comparison of children and adult brains. *IEEE Transactions on Medical Imaging*, 37(10):2165–2175, 2017.

Appendix A

Statement of Contribution

I confirm that the "Statement of Contribution to Doctoral Thesis Containing Publications (DRC16)", have been completed for each published article within the thesis, and are bound into the thesis and included in the electronic copy.

MASSEY UNIVERSITY
GRADUATE RESEARCH SCHOOL

STATEMENT OF CONTRIBUTION
DOCTORATE WITH PUBLICATIONS/MANUSCRIPTS

We, the candidate and the candidate's Primary Supervisor, certify that all co-authors have consented to their work being included in the thesis and they have accepted the candidate's contribution as indicated below in the *Statement of Originality*.

Name of candidate:	Rongyao Hu
Name/title of Primary Supervisor:	Associate Professor Xiaofeng Zhu
Name of Research Output and full reference:	
R.Hu, X. Zhu, Y. Zhu, and J. Gan. Robust SVM with adaptive graph learning. World Wide Web, 23(3): 1945-1968, 2020.	
In which Chapter is the Manuscript /Published work:	Chapter 3
Please indicate:	
• The percentage of the manuscript/Published Work that was contributed by the candidate:	85
and	
• Describe the contribution that the candidate has made to the Manuscript/Published Work:	
Designing study, carrying out the experiments and results, writing manuscript, and responding the reviewers' comments.	
For manuscripts intended for publication please indicate target journal:	
N/A	
Candidate's Signature:	Rongyao Hu 数字签名者：Rongyao Hu DN：cn=Rongyao Hu, o=Massey University, ou, email=r.hu@massey.ac.nz, c=NZ 日期：2022.03.30 11:36:51 +08'00'
Date:	30/03/2022
Primary Supervisor's Signature:	Xiaofeng Zhu 数字签名者：Xiaofeng Zhu 日期：2022.03.30 12:30:25 +08'00'
Date:	30/03/2022

(This form should appear at the end of each thesis chapter/section/appendix submitted as a manuscript/ publication or collected as an appendix at the end of the thesis)

MASSEY UNIVERSITY
GRADUATE RESEARCH SCHOOL

STATEMENT OF CONTRIBUTION
DOCTORATE WITH PUBLICATIONS/MANUSCRIPTS

We, the candidate and the candidate's Primary Supervisor, certify that all co-authors have consented to their work being included in the thesis and they have accepted the candidate's contribution as indicated below in the *Statement of Originality*.

Name of candidate:	Rongyao Hu
Name/title of Primary Supervisor:	Associate Professor Xiaofeng Zhu
Name of Research Output and full reference:	
R.Hu, Z. Peng, X. Zhu, J. Gan, Y. Zhu, J. Ma, and G. Wu. Muti-Band Brain Network Analysis for Functional Neuroimaging Biomarker Identification. IEEE Transactions on Medical Imaging, 40(12): 3843-3855, 2021.	
In which Chapter is the Manuscript /Published work:	Chapter 4
Please indicate:	
• The percentage of the manuscript/Published Work that was contributed by the candidate:	85
and	
• Describe the contribution that the candidate has made to the Manuscript/Published Work:	
Designing study, carrying out the experiments and results, writing manuscript, and responding the reviewers' comments.	
For manuscripts intended for publication please indicate target journal:	
N/A	
Candidate's Signature:	Rongyao Hu 数字签名者：Rongyao Hu DN：cn=Rongyao Hu, o=Massey University, ou, email=r.hu@massey.ac.nz, c=NZ 日期：2022.03.30 11:36:51 +08'00'
Date:	30/03/2022
Primary Supervisor's Signature:	Xiaofeng Zhu 数字签名者：Xiaofeng Zhu 日期：2022.03.30 12:31:00 +08'00'
Date:	30/03/2022

(This form should appear at the end of each thesis chapter/section/appendix submitted as a manuscript/ publication or collected as an appendix at the end of the thesis)

MASSEY UNIVERSITY
GRADUATE RESEARCH SCHOOL

STATEMENT OF CONTRIBUTION
DOCTORATE WITH PUBLICATIONS/MANUSCRIPTS

We, the candidate and the candidate's Primary Supervisor, certify that all co-authors have consented to their work being included in the thesis and they have accepted the candidate's contribution as indicated below in the *Statement of Originality*.

Name of candidate:	Rongyao Hu
Name/title of Primary Supervisor:	Associate Professor Xiaofeng Zhu
Name of Research Output and full reference:	
R.Hu, Z. Deng, and X. Zhu. Multi-scale Graph Fusion for Co-saliency Detection. In AAAI, pp. 1-9, 2021.	
In which Chapter is the Manuscript /Published work:	Chapter 5
Please indicate:	
• The percentage of the manuscript/Published Work that was contributed by the candidate:	85
and	
• Describe the contribution that the candidate has made to the Manuscript/Published Work:	
Designing study, carrying out the experiments and results, writing manuscript, and responding the reviewers' comments.	
For manuscripts intended for publication please indicate target journal:	
N/A	
Candidate's Signature:	Rongyao Hu 数字签名者：Rongyao Hu DN：cn=Rongyao Hu, o=Massey University, ou, email=r.hu@massey.ac.nz, c=NZ 日期：2022.03.30 11:36:51 +08'00'
Date:	30/03/2022
Primary Supervisor's Signature:	Xiaofeng Zhu 数字签名者：Xiaofeng Zhu 日期：2022.03.30 12:31:33 +08'00'
Date:	30/03/2022

(This form should appear at the end of each thesis chapter/section/appendix submitted as a manuscript/ publication or collected as an appendix at the end of the thesis)