Pediatric Bone Age Analysis and Brain Disease Prediction for Computer-Aided Diagnosis

Ibrahim Salim

A Thesis
in
The Concordia Institute
for
Information Systems Engineering

Presented in Partial Fulfillment of the Requirements
for the Ph.D. Degree in Information and Systems Engineering at
Concordia University
Montreal, QC, Canada

June 2022

CONCORDIA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

This is to certify that the thesis prepared

By: Ibrahim Ali Salim

Entitled: Pediatric Bone Age Analysis and Brain Disease Prediction for Computer-Aided Diagnosis

and submitted in partial fulfillment of the requirements for the degree of

Doctor Of Philosophy Information & Systems Engineering

complies with the regulations of the University and meets the accepted standards with respect to originality and quality.

Signed by the final examining committee:

_____ Chair
Dr. Ahmed Soliman

_____ Thesis Supervisor
Dr. Abdessamad Ben Hamza

_____ Thesis Supervisor
Dr.

_____ Thesis Supervisor
Dr.

_____ Examiner
Dr. Nizar Bouguila

_____ Examiner
Dr. Jamal Bentahar

_____ Examiner
Dr. Arash Mohammadi

_____ Examiner
Dr.

_____ External Examiner
Dr. Mohand Said Allili

Approved by

_____
Dr. Mohammad Mannan , Graduate Program Director

«Date» June 28,2022

_____
Dr. Mourad Debbabi , Dean

Abstract

Pediatric Bone Age Analysis and Brain Disease Prediction for Computer-Aided Diagnosis

Ibrahim Ali Salim, PhD

Concordia University, 2022

Recent advances in 3D scanning technology have led to a widespread use of 3D shapes in a multitude of fields, including computer vision and medical imaging. These shapes are, however, often contaminated by noise, which needs to be removed or attenuated in order to ensure high-quality 3D shapes for subsequent use in downstream tasks. On the other hand, the availability of large-scale pediatric hand radiographs and brain imaging benchmarks has sparked a surge of interest in designing efficient techniques for bone age assessment and brain disease prediction, which are fundamental problems in computer-aided diagnosis. Bone age is an effective metric for assessing the skeletal and biological maturity of children, while understanding how the brain develops is crucial for designing prediction models for the classification of brain disorders.

In this thesis, we present a feature-preserving framework for carpal bone surface denoising in the graph signal processing setting. The proposed denoising framework is formulated as a constrained optimization problem with an objective function comprised of a fidelity term specified by a noise model and a regularization term associated with data prior. We show through experimental results that our approach can remove noise effectively while preserving the nonlinear features of surfaces, such as curved surface regions and fine details. Moreover, recovering high quality surfaces from noisy carpal bone surfaces is of paramount importance to the diagnosis of wrist pathologies, such as arthritis and carpal tunnel syndrome. We also introduce a deep learning approach to pediatric bone age assessment using instance segmentation and ridge regression. This approach is comprised of two intertwined stages. In the first stage, we employ an image annotation and instance segmentation model to extract and separate different regions of interests in an image. In the second stage, we leverage the power of transfer learning by designing a deep neural network with

a ridge regression output layer. For the classification of brain disorders, we propose an aggregator normalization graph convolutional network by exploiting aggregation in graph sampling, skip connections and identity mapping. We also integrate both imaging and non-imaging features into the graph nodes and edges, respectively, with the aim of augmenting predictive capabilities. We validate our proposed approaches through extensive experiments on various benchmark datasets, demonstrating competitive performance in comparison with strong baseline methods.

Acknowledgments

I am deeply grateful to my supervisor Prof. A. Ben Hamza for his invaluable advice, continuous support, and patience during my PhD study. Thank you for your unwavering support and encouragement throughout my years of study, as well as the process of researching and writing this thesis. I have been tremendously lucky to have a supervisor who has given me the flexibility to pursue the research in which I am actually interested.

I want to thank my wife and my lovely daughters for supporting me throughout my study. Thank you for encouraging me in all of my pursuits and inspiring me to follow my dreams.

Last but not least, I want to express my heartfelt gratitude to my family for their support and hope. This thesis would not have been possible without that hope. Thank you all for your support and encouragement. I adore all of you!

Table of Contents

List of Tables

List of Figures

CHAPTER 1

Introduction

In this chapter, we start with the motivation behind this work, followed by the problem statement, objectives of the study, and literature review. Then, we present the basic preliminaries and background material, which include a brief overview of triangular mesh representation, regression analysis, convolutional neural networks, medical image segmentation, graph convolutional networks, and finally we conclude with the thesis contributions.

1.1 Framework and Motivation

As brands and content makers create more augmented reality experiences, the demand for tools to create digital 3D content has been growing at a faster pace over the past few years. With the proliferation of 3D scanners helping create 3D models, which are usually represented as triangle meshes, there is a rising need for robust mesh denoising techniques to remove inevitable noise in the measurements while preserving important surface features. Even with high-fidelity 3D scanners, the acquired 3D models are usually contaminated by noise, and therefore a reliable mesh denoising technique is often required. Motivated by the good performance of similarity-based image denoising methods [1], we design a geometric feature-preserving framework for carpal bone surface denoising using graph signal processing by leveraging a data-adaptive kernel similarity matrix in conjunction with a matrix balancing procedure.

On the other hand, bone age assessment using radiographic images is commonly employed in the diagnosis, treatment, monitoring of endocrine, genetic, and growth disorders in children. Using the Greulich-Pyleand and Tanner-Whitehouse [2, 3], physicians can manually assess radiographs

of hand bones. However, manual bone age assessment methods are time-consuming and rely heavily on radiologists' domain knowledge and experience. Recent bone age assessment methods employ deep neural networks to learn features from hand radiographs in an end-to-end fashion, achieving superior performance with substantially improved results in comparison with traditional approaches.

Understanding how the brain develops is essential for designing prediction models with the goal of classifying developmental disorders and degenerative neurological disorders characterized by impairments in language, learning, behavior, and physical parts of life. These disorders affect the nervous system and often occur when nerve cells in the brain or peripheral nervous system lose function over time and ultimately die. By exploring the brain networks' connectivity, which represents the relationship between the brain regions of interest, we can better understand the pathological underpinnings of neurological disorders using graph representation learning in an effort to classify diseases automatically without manual feature extraction and selection [4].

1.2 Problem Statement

In this thesis, we briefly describe the carpal bone surface denoising, pediatric bone age assessment, and brain disorder prediction problems.

1.2.1 Carpal Bone Surface Denoising

During the acquisition stage, 3D shapes are often contaminated by noise. The main problem in 3D shape denoising is how we can distinguish between noise and features, especially sharp surface features. The noise-induced degradation model is usually represented as $\mathbf{v} = \mathbf{u} + \boldsymbol{\eta}$, where $\mathbf{v}$ is the observed graph signal, $\mathbf{u}$ is the original, noise-free graph signal, and $\boldsymbol{\eta}$ is a random noise. The noise process is often assumed to be Gaussian distributed with zero mean and same variance. The goal of surface denoising is to recover the noise-free graph signal $\mathbf{u}$ from the observed graph signal $\mathbf{v}$ with some *a priori* knowledge about the distribution of the noise process.

1.2.2 Pediatric Bone Age Assessment

Bone age assessment is vital for the diagnosis and treatment of children with suspected growth disorders. Traditional bone age assessment methods rely heavily on expert radiologists and often suffer from significant inter- and intra-observer variability. The recent development of convolutional neural networks and their variants has provided a powerful tool to make accurate prediction of bone age with the aim of evaluating the biological maturity of children. Bone age assessment is

typically achieved by extracting regions-of-interest features from pediatric hand images, followed by employing a deep learning model to estimate the bone age from these radiographs.

1.2.3 Brain Disorder Prediction

Graphs are a prevalent data representation in many real-world applications, including social networks, biological protein-protein interaction networks, molecular graph structures, and brain connectivity networks. The human brain can be seen as an graph, where each node in the graph represents a region of interest with an associated node features vector consisting of imaging data, and each edge represents a pairwise similarity between two adjacent nodes with an edge feature vector comprised of non-imaging data. While graph convolution based methods have become ubiquitous in graph representation learning, their application to disease prediction problems has not been sufficiently explored, especially in the prediction of neurodevelopmental and neurodegenerative brain disorders

Given the labels of a subset of the graph nodes, the goal of semi-supervised learning for brain disease or disorder prediction is to predict the unknown labels of the remaining nodes. For a binary classification problem, the label of each node i in the labelled set $\mathcal{D}_l$ can be represented as a C-dimensional one-hot vector $\mathbf{y}_i \in \{0,1\}^C$, where C is the number of classes with 0 and 1 representing "healthy" and "diseased" status of the subjects, respectively.

1.3 Objectives

In this thesis, we aim to achieve the following objectives:

- We present a geometric framework for carpal bone surface denoising with the objective of removing undesirable noise while preserving salient features [5]. The proposed surface denoising framework is formulated as a constrained minimization problem, which can be solved efficiently using the conjugate gradient method.

- We design a deep neural network architecture by leveraging instance segmentation and ridge regression with the aim of predicting pediatric bone age [6]. Bone age assessment is often used in clinical practice by pediatricians for the children's skeletal maturation assessment in an effort to quantify the difference or gap between a child's bone age and the associated chronological age.

- We propose an aggregator normalization graph convolutional network for with the goal of classifying neurodevelopmental and neurodegenerative brain disorders. The objective is to

build a model that can learn discriminative graph node representations by incorporating both imaging and non-imaging features into graph nodes and edges, respectively, in order to improve predictive capabilities.

1.4 Literature Review

Carpal Bone Surface Denoising. In human anatomy, the wrist, also called carpus, is a complex joint that connects the hand to the forearm. It is composed of eight carpal bones, located between the five metacarpal bones of the hand and the radius and ulna bones of the forearm. Each carpal bone has a distinct shape and plays a crucial functional role in the wrist stability and motion. Carpal bones are relatively prone to injury, as force or stress can injure any of the bones. The most commonly injured carpal bone in the human wrist is the scaphoid bone, located near the base of the thumb. The human wrist can be rendered using triangular mesh models of the cortical bone surface via segmentation of a computerized tomography (CT) volumetric image. Triangle meshes offer a simple and flexible way to represent and handle complex geometric shapes. However, the surface of a triangular mesh model reconstructed from real-world data is often corrupted by noise. Hence, it is critical to develop effective surface denoising techniques to attenuate the inevitable noise in measurements in order to ensure high-quality 3D shapes for use in downstream tasks. The Laplacian mesh filtering flow is one of the simplest surface denoising methods, as it repeatedly and simultaneously adjusts the location of each mesh vertex to the geometric center of its neighboring vertices [7], but it tends to oversmooth the surface features.

Existing approaches to surface denoising can be divided into two main groups, namely isotropic and anisotropic, based on how they treat noise and salient features. Wang *et al.* [8] introduce a mesh denoising framework using a combination of bilateral filtering, feature detection, anisotropic neighborhood searching, surface fitting and projection techniques. The mesh features are detected and classified into non-feature vertices and feature vertices. Then, the corresponding anisotropic neighborhoods for each vertex are searched by constructing a weighted dual graph. Zhu *et al.* [9] develop a mesh denoising method by filtering each face normal within its piecewise smooth region in lieu of using the anisotropic neighborhood. This is done by classifying mesh faces into several types using a face normal tensor voting and then performing a normal filter to obtain a denoised coarse normal field. Fleishman *et al.* [10] propose a bilateral mesh denoising method that utilizes local neighborhoods to filter each mesh vertex in the normal direction. Zheng *et al.* [11] present an anisotropic mesh denoising framework via normal field denoising by considering normals as a surface signal defined over the original mesh, as well as designing a bilateral normal filter. The

bilateral updating is cast as an optimization problem with an objective function defined as weighted combination a smoothness term and a data term, where the weight parameter can be adjusted to control the amount of denoising. Zhang *et al.* [12] design a two-stage mesh denoising framework by applying a joint bilateral filter to the face normals and constructing a guidance normal field that indicates surface features in the presence of noise, followed by updating the vertex positions according to the filtered face normals.

Bone Age Assessment. Bone age, also called skeletal age, is a widely used measure for estimating the maturity of a child's skeletal system by taking an X-ray image (radiograph) of the wrist and comparing it to a standardized reference from an atlas. Assessment of bone age is vital for the diagnosis and treatment of children with suspected growth disorders [13], as the difference or gap between a child's bone age and their chronological age might indicate a growth problem. The objective of pediatric bone age assessment is to evaluate growth and maturity, as well as to diagnose and manage pediatric disorders. The most commonly used methods for bone age assessment in clinical practice are the Greulich-Pyleand and Tanner-Whitehouse [2, 3], which involve left hand and wrist radiographs due in large part to the fact that most people are right-handed, and consequently, the right hand is more likely to be injured than the left hand. The Greulich-Pyleand method is an atlas method in which bone age is evaluated by comparing the radiograph of the patient with the nearest standard radiograph in the atlas, whereas the Tanner-Whitehouse method is a scoring approach that relies on the systematic evaluation of the maturity of all the bones in the hand and wrist. However, bone age assessment using the Tanner-Whitehouse method requires a longer time than the Greulich-Pyleand method. Moreover, these traditional bone age assessment methods are time consuming, rely on trained radiologists, and suffer from significant inter- and intra-observer variability. To address these limitations, several automated methods have recently been developed, including BoneXpert [14], an automated technique for assessing bone age and bone density expressed in the Bone Health Index using conventional hand X-rays.

More recently, several deep transfer learning models have been proposed to tackle the bone age assessment problem. Van Steenkiste *et al.* [15] propose an automated bone age assessment method to assist physicians by combining a pre-trained deep convolutional neural network with Gaussian process regression with the aim of aggregating the predictions for rotated and flipped versions of the same radiograph in order to increase overall predictive performance. Iglovikov *et al.* [16] present a deep learning approach for bone age assessment using a multi-step preprocessing pipeline. This pipeline includes background removal by segmenting the image of the hand using a modified version of the U-Net architecture, contrast normalization and detection of key points, the application of affine transformations to register segmented images in a common coordinate space,

and bone age regression and classification using pre-trained convolutional neural networks that are trained by minimizing the mean absolute error. Inspired by the traditional Tanner-Whitehouse method, Wu *et al.* [17] introduce a deep learning framework for simultaneous hand segmentation and bone age assessment. This method is comprised of of two subnetworks: a Mask region-based convolutional neural network (Mask R-CNN) subnetwork for pixel-wise hand segmentation from X-ray images to avoid the distractions of other projects, and a residual attention network for hand bone age assessment that forces the network to automatically attend to important regions. Son *et al.* [18] propose an automated bone age assessment system comprised of the extraction of regions of interest and classification of the skeletal maturity levels of the regions by converting them into scores, which in turn are used in conjunction with the correlation matrix to predict the bone age.

Brain Disorder Prediction. Graph-structured data is prevalent real-world applications, including social networks, biological protein-protein interaction networks, molecular graph structures, traffic and transportation networks, and brain connectivity networks. The brain network has a complex structure that can be modeled as a graph comprised of the brain's regions of interest as nodes and their connectivity as edges. Understanding how the brain develops is vital for designing prediction models on graph-structured data with the aim of classifying developmental disorders and degenerative neurological disorders such as autism spectrum disorder and Alzheimer's disease [19, 20]. Autism spectrum disorder is a neurodevelopmental condition related to brain development that impacts how a person perceives and socializes with others, causing problems in social interaction and communication. It begins in early childhood and affects approximately 1% of the global population, with males being approximately four times more susceptible than females. Alzheimer's disease is a progressive neurodegenerative disorder that primarily affects the elderly population. Resting-state functional magnetic resonance imaging has enabled clinicians to better identify the pathophysiology of brain disorders. The ability to distinguish between autism spectrum disorder, Alzheimer's disease and normal control individuals help in the characterization of underlying causes, resulting in better diagnosis and treatment.

Graph convolutional networks (GCNs) and their variants have recently become the method of choice in graph representation learning due to their ability to capture the graph structure [21]. Xu *et al.* [22] propose a graph wavelet neural network, which is a GCN-based architecture that employs spectral graph wavelets instead of graph Fourier bases to define a graph convolution. Zeng *et al.* [23] present a graph sampling-based learning method that samples the training graph rather than nodes or edges across the GCN layers, as well as an aggregator normalization technique for eliminating bias in minibatch estimates. Chen *et al.* [24] introduce an extension of the GCN model that employs skip connections from the input layer and identity mapping with the learnable weight

matrix of each layer in order to mitigate the over-smoothing problem, which occurs when node representations become indistinguishable as the network depth increases, resulting in a drop in performance.

More recently, GCNs have demonstrated significant promise in computer-assisted diagnosis, particularly in the prediction of brain disorders such as autism spectrum disorder and Alzheimer's disease [4,25–31]. Parisot *et al.* [4] employ spectral GCNs to tackle the disease prediction problem on population graphs by leveraging both imaging and non-imaging information in order to boost classification performance. Inspired by the successful inception architecture for convolutional neural networks, Kaiz *et al.* [25] propose InceptionGCN, a disease prediction model that leverages spectral convolutions with different kernel sizes, resulting in improved performance over regular GCN architectures due to its ability to capture the local and global context of heterogeneous graph structures. Cosmo *et al.* [26] present an end-to-end trainable graph learning architecture for disease prediction and classification using latent-graph learning with the aim of performing node classification. The classification model is comprised of several graph convolutional layers, followed by a fully connected layer to predict the patient label. Cao *et al.* [27] introduce a DeepGCN framework for disease prediction on population graphs by integrating residual networks and a DropEdge strategy in order to help avoid the vanishing gradient, over-fitting, and over-smoothing problems. Wen *et al.* [31] propose a prior brain structure learning-guided multi-view graph convolutional neural network for autism spectrum disorder diagnosis by exploiting graph structure learning and multi-task graph embedding learning in an effort to improve classification performance and identify potential functional subnetworks.

1.5 Preliminaries

In this section, we present a terse overview of triangular mesh representation, conjugate gradient method, regression analysis, convolutional neural networks, transfer learning, medical image segmentation, graph representation learning, graph convolutional networks, and brain connectivity networks.

1.5.1 Triangular Mesh Representation

A triangle mesh $\mathbb{M}$ can be defined as a graph $\mathbb{G} = (\mathcal{V}, \mathcal{E})$ or $\mathbb{G} = (\mathcal{V}, \mathcal{T})$, where $\mathcal{V} = \{\mathbf{v}_1, \ldots, \mathbf{v}_n\}$ is a set of vertices, $\mathcal{E} = \{\mathbf{e}_{ij}\}$ is a set of edges, and $\mathcal{T} = \{\mathbf{t}_1, \ldots, \mathbf{t}_n\}$ is a set of triangles, as shown in Figure 1.1. Each edge $\mathbf{e}_{ij}$=$[\mathbf{v}_i, \mathbf{v}_j]$ connects two adjacent vertices $\mathbf{v}_i$ and $\mathbf{v}_j$, which are usually

denoted by $\mathbf{v}_i \sim \mathbf{v}_j$ or simply $i \sim j$. The neighborhood of a vertex $\mathbf{v}_i$, denoted by $\mathcal{N}_i$, is the set

$$\mathcal{N}_i = \{\mathbf{v}_j \in \mathcal{V} : \mathbf{v}_j \sim \mathbf{v}_i\}. \tag{1.1}$$

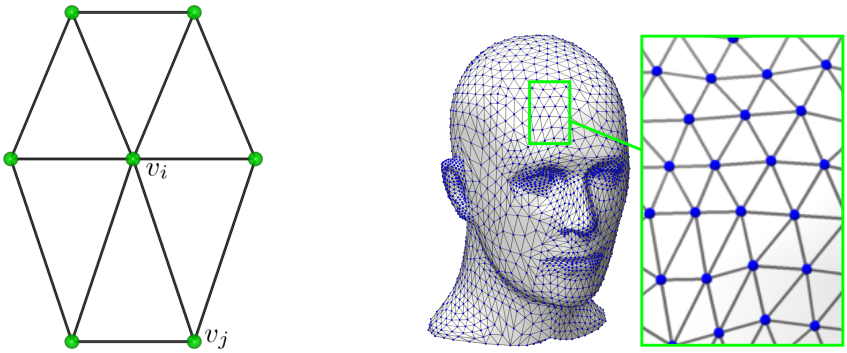

Figure 1.1: Vertex neighbors (left) and triangular mesh representation (right).

1.5.2 Conjugate Gradient Method

The conjugate gradient method is one of the most popular iterative algorithms for solving sparse systems of linear equations

$$\mathbf{A}\mathbf{x} = \mathbf{b}, \tag{1.2}$$

where $\mathbf{x}$ is an unknown vector, $\mathbf{b}$ is a known vector, and $\mathbf{A}$ is $n \times n$ symmetric positive-definite matrix. Finding the solution $\mathbf{x}^\star$ to this system of linear equations is equivalent to minimizing the following quadratic function

$$f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^\intercal \mathbf{A}\mathbf{x} - \mathbf{x}^\intercal \mathbf{b} \tag{1.3}$$

and setting its gradient to zero, i.e. $\nabla f(\mathbf{x}^\star) = \mathbf{A}\mathbf{x}^\star - \mathbf{b} = 0$. The uniqueness of this solution is guaranteed since the Hessian matrix $\nabla^2 f(\mathbf{x}) = \mathbf{A}$ is a symmetric positive-definite matrix. The main algorithmic steps of the conjugate graduate method are summarized in Algorithm 1.

1.5.3 Regression Analysis

Regression analysis is a statistical technique for investigating the relationships between variables. The goal of regression analysis is to determine the values of parameters for a function that cause the function to best fit a given set of data observations.

Multiple Linear Regression. The objective of multiple linear regression is to model the linear relationship between the explanatory (independent) variables and response (dependent) variables

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}, \tag{1.4}$$

Algorithm 1 Iterative Conjugate Gradient

Input initial point $\mathbf{x}_0$

$\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0, \quad \mathbf{p}_0 = \mathbf{r}_0$
while $\mathbf{r}_k \neq 0$ **do**
  Construct the step size $\alpha_k = \frac{\mathbf{r}_k^\intercal \mathbf{r}_k}{\mathbf{p}_k^\intercal \mathbf{A}\mathbf{p}_k}$
  Construct next iteration $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k$
  Construct new residual $\mathbf{r}_{k+1} = \mathbf{r}_k + \alpha_k \mathbf{A}\mathbf{p}_k$
  Construct scalar for linear combination for next direction $\beta_k = \frac{\mathbf{r}_{k+1}^\intercal \mathbf{r}_{k+1}}{\mathbf{r}_k^\intercal \mathbf{r}_k}$.
  Construct next conjugate vector $\mathbf{p}_{k+1} = \mathbf{r}_{k+1} + \beta_k \mathbf{p}_k$
  $k = k + 1$
end while

where $\mathbf{y}$ is the dependent or response vector, $\mathbf{X}$ is the independent or predictor data matrix, $\boldsymbol{\beta}$ is an unknown vector of regression parameters to be estimated, and $\boldsymbol{\varepsilon}$ is a random error vector with unknown variance. The rows of the predictor data matrix $\mathbf{X}$ correspond to observations, and its columns correspond to predictor variables. The vector $\boldsymbol{\beta}$ of unknown parameters can be estimated using least-square estimation by minimizing the following sum of squared deviations

$$\hat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}} \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2, \tag{1.5}$$

and it is given by $\hat{\boldsymbol{\beta}} = (\mathbf{X}^\intercal \mathbf{X})^{-1} \mathbf{X}^\intercal \mathbf{y}$.

Ridge Regression. Ridge regression is a regularized regression method that aims to minimize the following objective function

$$\hat{\boldsymbol{\beta}}_R = \arg\min_{\boldsymbol{\beta}} \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 + \lambda \|\boldsymbol{\beta}\|_2, \tag{1.6}$$

where $\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2$ is the residual sum of squares, $\|\boldsymbol{\beta}\|_2^2$ is the L_2-penalty (regularization) term, λ is a regularization parameter that decides the shrinkage amount of the parameters. A larger value of λ indicates more shrinkage.

The solution to the minimization problem for ridge regression is given by

$$\hat{\boldsymbol{\beta}}_R = (\mathbf{X}^\intercal \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^\intercal \mathbf{y} \tag{1.7}$$

Ridge regression has been shown to improve the least-squares estimate when multicollinearity is present. Using ridge regression, we can reduce overfitting and ensure achieving the generalization by penalizing the estimates of the parameters. If these estimates are very high, the residual sum of squares term will be reduced, but the penalty term will increase.

LASSO Regression. The Least Absolute Shrinkage and Selection Operator (LASSO) regression is a regularized regression technique that encourages sparse models by minimizing the following

objective function

$$\hat{\boldsymbol{\beta}}_L = \arg\min_{\boldsymbol{\beta}} \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 + \lambda\|\boldsymbol{\beta}\|_1, \tag{1.8}$$

where $\|\boldsymbol{\beta}\|_1$ is the L_1-penalty (regularization) term that induces sparsity and λ is a regularization parameter that controls the amount of shrinkage. As the value of λ increases, more parameters are shrunk to zero and hence eliminated, meaning that the number parameters decreases.

1.5.4 Convolutional Neural Networks

A Convolutional Neural Network (CNN) is a deep architecture inspired by the way humans process visual information [32]. The architecture of a CNN resembles the connectivity pattern of neurons in the human brain, and makes use of feedforward artificial neural networks in which individual neurons are tiled in such a way that they respond to overlapping regions in the visual field. CNNs are comprised of multiple layers that can be categorized into three types: convolutional, subsampling (pooling), and fully-connected. A convolutional layer consists of a rectangular grid of neurons, and applies a set of filters that process small local parts of the input where these filters are replicated along the whole input space. Each neuron takes inputs from a rectangular section of the previous layer; the weights for this rectangular section are the same for each neuron in the convolutional layer. Thus, the convolutional layer is just an image convolution of the previous layer, where the weights specify the convolution filter. A subsampling (pooling) layer takes small rectangular blocks from the convolutional layer and subsamples it with with average or max pooling to produce a single output from that block. This adds translation invariance and tolerance to minor differences of positions of objects parts. Higher layers use more broad filters that work on lower resolution inputs to process more complex parts of the input. Similar to a feedforward neural network, a fully-connected layer takes all neurons in the previous layer and connects them to each of its neurons. A CNN architecture with two convolutional layers and two subsampling layers is illustrated in Figure 1.2.

A CNN architecture is typically composed of an input layer, an output layer, and several hidden layers in between. These layers perform operations that alter the data with the objective of learning features specific to the data. The most common layers are convolution, activation, pooling, and fully-connected.

Convolutional Layers. A convolutional layer is the main building block of a CNN architecture, and contains a set of filters (kernels) that are convolved with an image, performing element-wise multiplications in order to create feature maps that summarize the presence of detected features in the image. A convolution puts the input images through a set of convolutional filters, each of which activates certain features from the images. Only a small region of neurons in the input image

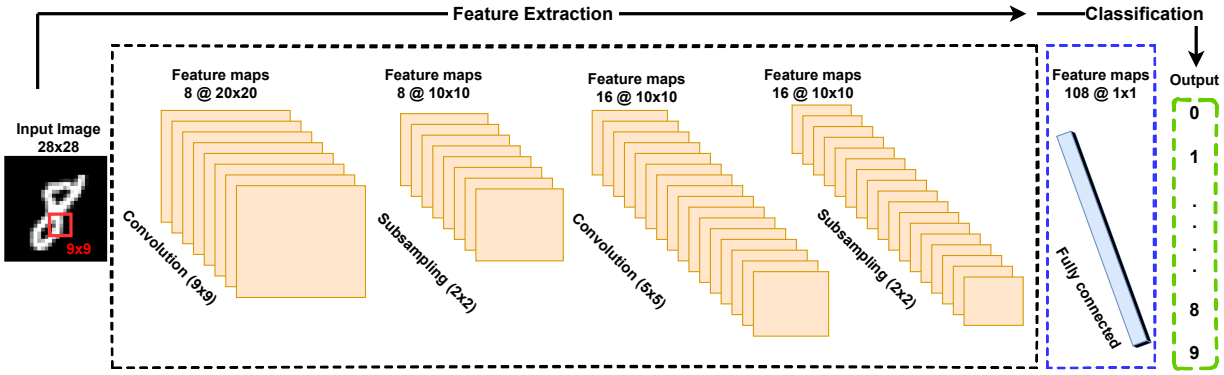

Figure 1.2: Basic architecture of a convolutional neural network.

connects to neurons in the hidden layer in a convolutional neural network. The region or filter is a small spatially, also referred to as a kernel, is applied across the input.

Local Receptive Field. The kernels in the input image are referred to as local receptive fields. Each connection in the filter learns a weight, representing the local receptive field of the input image.

Weight Sharing. Convolutional layers employ a weight sharing approach to reduce the number of parameters (weights and biases) that need to be learned. Every neuron in the hidden layer shares the same weights of the local receptive field. The purpose of weight sharing is to not only to reduce the model training time and cost, but also to make feature learning insensitive to feature location in the image.

Feature Maps. A filter slides over the input image (convolution operation) to yield a feature map. Three parameters determine the size of the feature map: depth, stride, and zero-padding. Depth is the number of filters used in the convolution operation, where each depth learns a different feature. Stride refers to the number of pixels by which we slide the filter across the input volume. When the stride is set to 1, we move the filters one pixel at a time. Zero-padding refers to the process of padding the input image with zeros around the border with the aim of applying the filter to bordering elements of the image. The size of the filter (kernel) is usually an odd number so that it has a central pixel.

Pooling Layers. Pooling layers are typically placed after one or more convolutional layers. Spatial pooling (also called subsampling or downsampling) reduces the dimensionality of each feature map and retains the most important information of an image. Spatial pooling can be broadly classified into three different types: Max, Average, and Sum. In practice, max-pooling has been shown to yield better results. Similar to the convolutional layer, the pooling layer is responsible for reducing the spatial size of the convolved feature map.

Fully-Connected Layers: A fully-connected layer is typically a Multi Layer Perceptron (MLP) that uses a softmax classifier in the output layer. The term "fully connected" means that every neuron in the previous layer is connected to every neuron on the next layer. The output from the convolutional and pooling layers represent high-level features of the input image. The purpose of the fully-connected layer is to use these features for classifying the input image into various classes based on the training dataset. Also, adding a fully-connected layer helps in learning non-linear combinations of these features.

Activation Function. The purpose of the activation function is to introduce non-linearity into the output of a neuron in a deep neural network. It decides whether a neuron should be activated or not, and hence only the activated features are carried forward into the next layer. The reason behind applying a non-linear activation function is that convolution is a linear operation. A commonly used activation function is the Rectified Linear Unit (ReLU) defined as

$$\text{ReLU}(x) = \max(0, x), \tag{1.9}$$

where x is the input to a neuron. Unlike other activation functions, ReLU helps in solving the vanishing gradient problem and also allows for faster and more effective training by mapping negative values to zero and maintaining positive values.

1.5.5 Transfer Learning

Transfer learning refers to the transfer of knowledge from one learned task to a new task. To this end, we take learned features from a pretrained network and transfer them to a new problem. The vast majority of the pretrained networks are trained on a subset of the ImageNet database, which is used in the ImageNet Large-Scale Visual Recognition Challenge. These networks have been trained on ImageNet (a dataset of 1.2 million images) and can classify images into 1000 object categories. Using a pretrained network with transfer learning is typically much faster and easier than training a network from scratch.

Visual Geometry Group (VGG) Networks. VGG is a family of very deep convolutional networks (up to 19 weight layers), which have shown to achieve state-of-the-art accuracy on the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) classification and localization tasks [33]. VGG networks use very small convolutional 3×3 kernels with padding of 1 and max-pooling of size 2×2 throughout the entire network. Each convolutional layer is followed by a ReLU activation function. The most popular configurations of VGG networks are VGG-16 and VGG-19, which are 16 and 19 layers deep, respectively. For instance, the VGG-16 network consists of 16 layers with learnable parameters: 13 convolutional layers and 3 fully-connected layers.

The first two fully-connected layers have 4096 channels each, and the last layer has 1000 channels, 1 for each class. The input to any of the network configurations is considered to be a fixed size $224 \times 224 \times 3$ RGB image. The only pre-processing done is normalizing the RGB values for every pixel. This is done by subtracting the mean value from every pixel. VGG-16 is relatively large with approximately 138 million parameters. Figure 1.3 shows the architecture of the VGG-16 network.

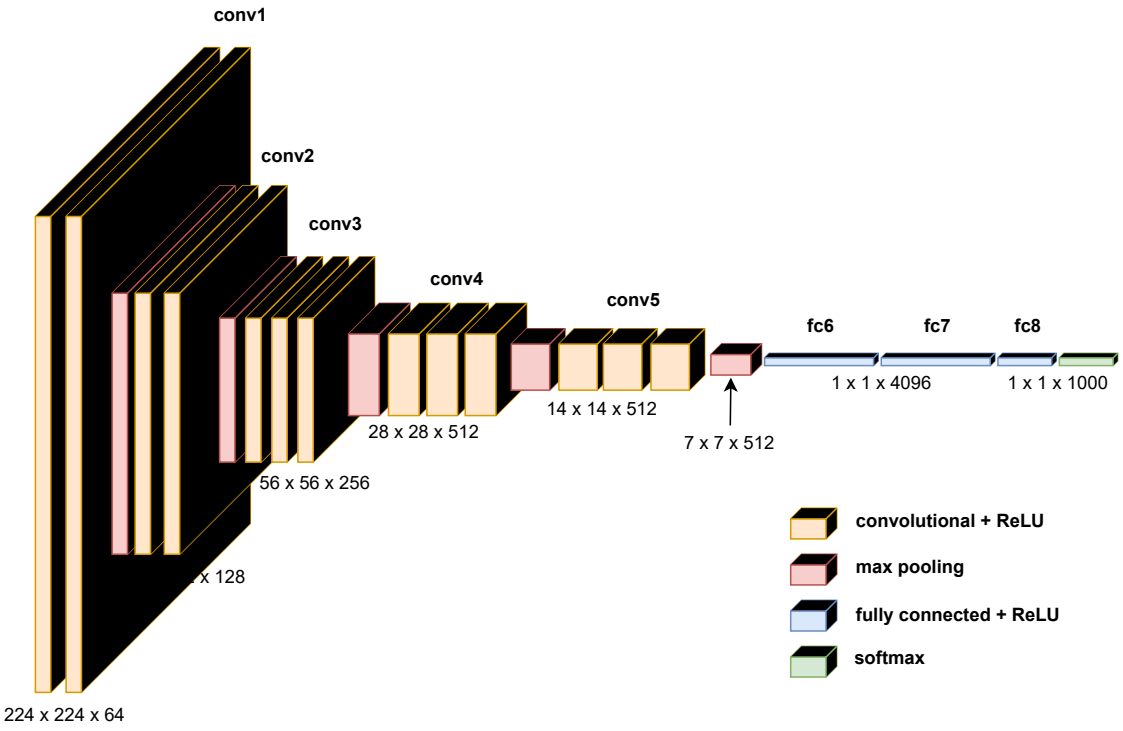

Figure 1.3: VGG-16 architecture.

Residual Networks (ResNets). While deep neural networks have proven effective in image classification, they suffer, however, from the vanishing gradient, as well as from the degradation problem, i.e. as the network depth increases, accuracy becomes saturated and then degrades rapidly. This means that stacking additional layers in a deep neural network results in higher training error. To tackle these issues, Kaiming *et al.* [34] proposed residual networks, a family of deep convolutional neural networks that are comprised of residual blocks to improve the model accuracy using the concept of skip connections to jump over some layers. Skip connections help alleviate the problem of the vanishing gradient in deep neural networks by allowing this alternate shortcut path for the gradient to flow through. They also allow the model to learn the identity mapping, which ensures that the higher layer will perform at least as good as the lower layer, and not worse. Consider $\mathcal{H}(\mathbf{x})$ to be an underlying mapping to be fitted by a few stacked layers, with $\mathbf{x}$ representing the inputs to the first of these layers. If one hypothesizes that many nonlinear layers may asymptotically

approximate complicated functions, then is similar to hypothesizing that they can also asymptotically approximate residual functions, i.e., $\mathcal{H}(\mathbf{x}) - \mathbf{x}$. Thus, instead of expecting stacked layers to approach $\mathcal{H}(\mathbf{x})$, we explicitly allow them to approximate a residual function $\mathcal{F}(\mathbf{x}) := \mathcal{H}(\mathbf{x}) - \mathbf{x}$. An illustration of residual learning is shown in Figure 1.4. A common configuration of residual networks is ResNet-50, which is a convolutional neural network that is 50 layers deep. This pre-trained network can classify images into 1000 object categories, and hence the network has learned rich feature representations for a wide range of images.

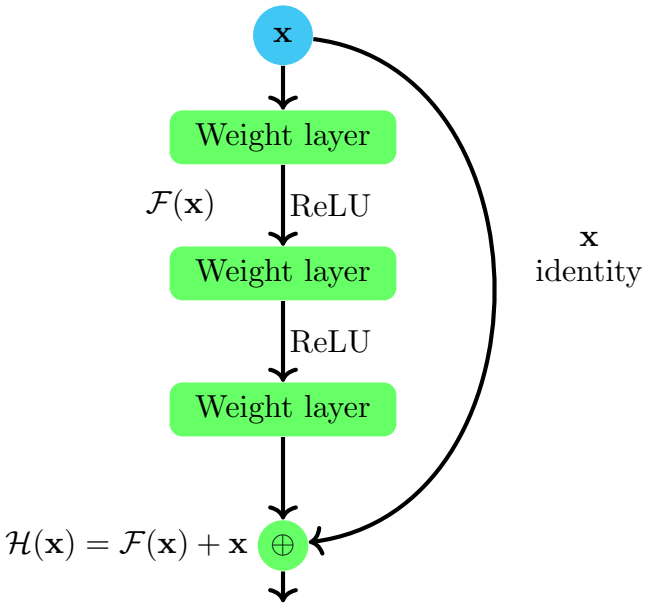

Figure 1.4: Residual learning of a 3-layer block.

1.5.6 Medical Image Segmentation

Image segmentation is a commonly used technique in image processing and computer vision to partition an image into multiple parts or regions, often based on the characteristics of the pixels in the image. It helps in understanding the image at a much lower level (e.g., the pixel level). The task of medical image segmentation is to label each pixel of an object of interest in medical images, and is often used in clinical applications such as computer-aided diagnosis [35]. Recent state-of-the-art medical image segmentation methods can be broadly classified into two main categories: semantic segmentation and instance segmentation. The former refers to the process of partitioning an image into semantically meaningful parts, and the goal is to assign a label to every pixel in the image. The latter yields an object level representation by assigning instance labels to each object pixel [36].

Deep neural networks, and in particular convolutional neural networks, have been successfully applied to image segmentation. Two representative semantic and instance segmentation methods are the U-Net network and mask R-CNN network, respectively.

U-Net Model. The U-Net architecture, built upon the fully convolutional network, has proven to be effective in biomedical image segmentation [16, 37–39]. As shown in Figure 1.5, U-Net is composed of an encoder subnetwork (contracting path) for capturing context by encoding the input image into low-level feature representations at multiple levels and a decoder subnetwork (expansive path) for semantically projecting these feature representations into the pixel space in an effort to enable precise localization via transposed convolutions. Both the contracting path and the expansive path make up 23 convolutional layers, each is followed by a ReLU activation function.

The encoder resembles the typical architecture of a convolutional neural network and consists of two convolution layers, followed by ReLU activation function and max pooling for each convolution layer to capture the context of the input image in order to perform segmentation. The convolutional layer applies a filter to an input to create a feature map that summarizes the presence of extracted features from the input, whereas the pooling layer downsamples or reduces the size of each feature map by a factor of 2.

On the other hand, the decoder, which also consists of five blocks, may be regarded as an operator that performs the reverse of the downsampling path. Every block in the expanding path comprises an up-convolution (i.e. upsampling the features), followed by two convolutions with ReLU activations. The expanding path aims to provide precise estimation of localization combined with contextual information from the contracting path. The goal of using U-Net is to convert an image into a feature vector and then reconstruct the image from that vector. More precisely, U-Net utilizes the same feature maps that are used for contraction to expand a vector to a segmented image.

Mask R-CNN Model. Instance Segmentation is a concept closely related to object detection. However, unlike object detection the output is a mask (or contour) containing the object instead of a bounding box. Unlike semantic segmentation, we do not label every pixel in the image, but rather we are interested only in finding the boundaries of specific objects. In other words, instance segmentation identifies each instance of each object in an image, and differs from semantic segmentation in that it does not categorize every pixel.

Mask Region based Convolutional Neural Network (Mask R-CNN) is an instance segmentation technique, which locates each pixel of every object in the image instead of the bounding boxes, and it is an extension of Faster R-CNN [40]. The input of Mask R-CNN is an image and the output is a bounding box and a mask that segment each object in the image, as shown in Figure 3.2. Mask

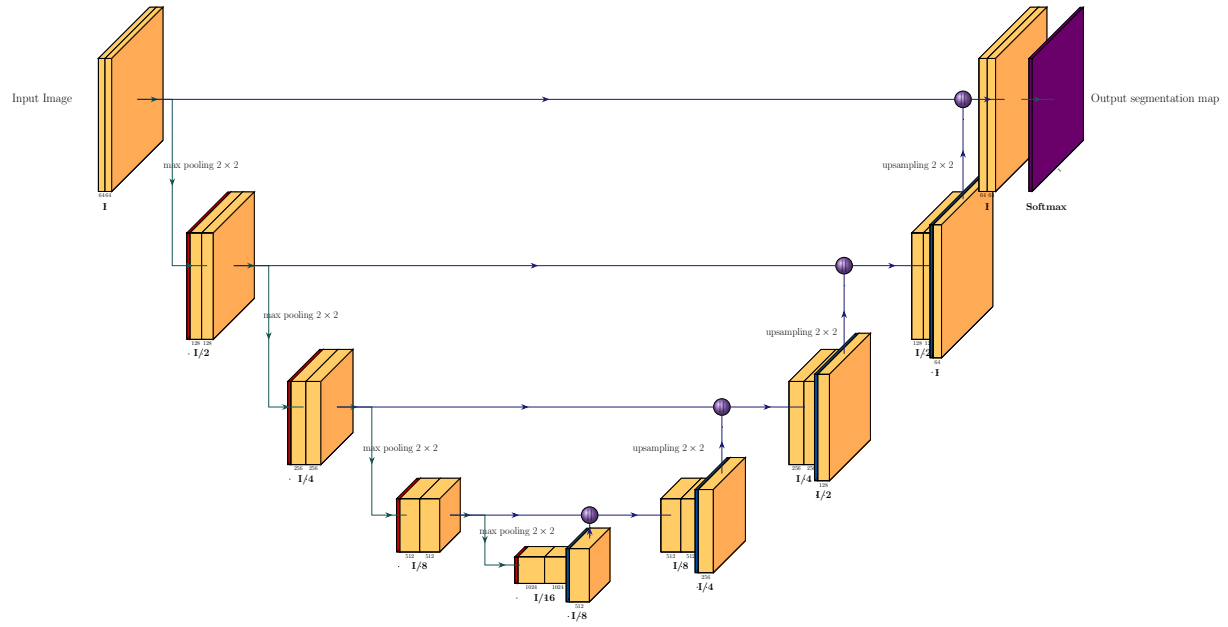

Figure 1.5: U-Net architecture for semantic segmentation.

R-CNN is comprised of two main stages: region proposals and then classifying the proposals and generating bounding boxes and masks. It does so by using an additional fully convolutional network on top of a convolutional neural network based feature map with input as feature map and gives matrix with 1 on all locations where the pixel belongs to the object and 0 elsewhere as the output.

- **Phase 1:** The initial phase includes two networks: a backbone network such as ResNet or VGG network to extract the features and a region proposal network. These networks are performed once per image to provide a collection of region proposals. Region suggestions are feature map regions that contain the object.

- **Phase 2:** In this phase, the network predicts bounding boxes and object class for each proposed region generated in Phase 1. Each proposed region might be of varying size, whereas fully connected layers in networks always require a constant size vector to generate predictions. The size of these proposed regions is determined by either the RoI pool (quite similar to max-pooling) or the RoIAlign layer.

Using Mask R-CNN we can generate pixel-wise masks for each object in an image, thereby allowing us to segment the foreground object from the background. The key point is to decouple the classification and the pixel-level mask prediction tasks. The mask branch is a small fully-connected network applied to each RoI, predicting a segmentation mask in a pixel-to-pixel manner. Classification and boundary box predictions Top final predictions from Mask R-CNN Because pixel-level

segmentation requires much more fine-grained alignment than bounding boxes, mask R-CNN improves the RoIAlign layer so that RoI can be better and more precisely mapped to the regions of the original image. The output of the RoIAlign layer is then passed into the Mask head, which is comprised of two convolution layers. It builds a mask for each RoI, segmenting an image pixel by pixel.

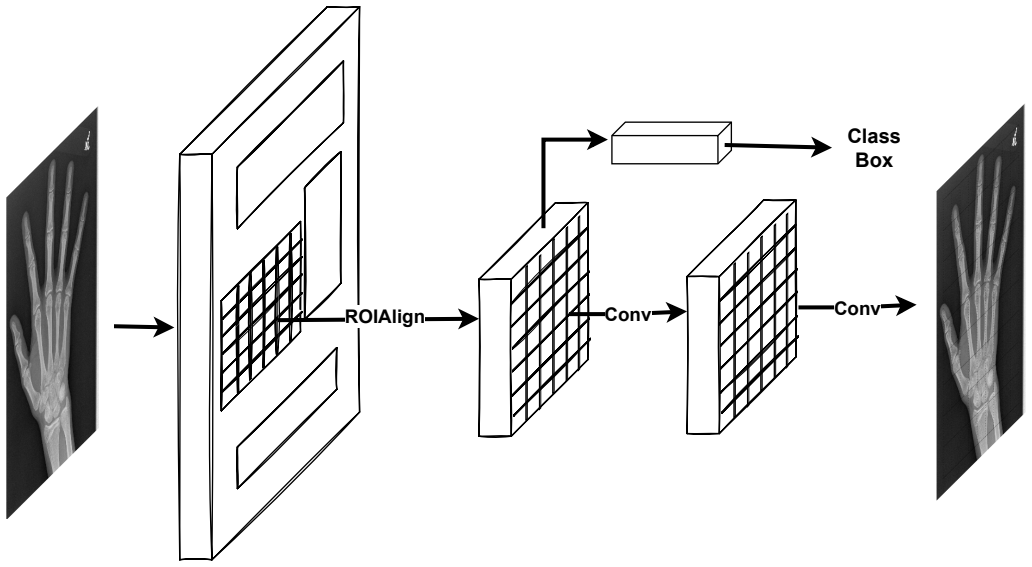

Figure 1.6: Mask R-CNN framework of instance segmentation.

1.5.7 Graph Representation Learning

Basic Notions. A graph $\mathbb{G} = (\mathcal{V}, \mathcal{E})$ is defined as the sets of nodes (or vertices) $\mathcal{V}$ and edges $\mathcal{E}$. The number of nodes in a graph is denoted by $N = |\mathcal{V}|$, and the number of edges by $M = |\mathcal{E}|$. We denote by $\mathbf{A} = (\mathbf{A}_{ij})$ an $N \times N$ adjacency matrix (binary or real-valued) whose (i, j)-th entry $\mathbf{A}_{ij}$ is equal to the weight of the edge between neighboring nodes i and j; and 0 otherwise.

The goal of graph representation learning (or graph embedding) is to map each node in the graph to a vector in a low-dimensional vector space while preserving the structure of the original graph. The resulting nonlinear and highly informative graph embeddings (or features) can then be used as input to machine learning models for various downstream tasks such as node classification, link prediction, clustering, recommendation, and graph classification. Given a graph $\mathbb{G}$, the aim of graph representation learning is to learn a mapping $f : \mathcal{V} \to \mathbb{R}^d$, where $d \ll N$ is the dimension of the embedding. An illustration of node embedding is shown in Figure 1.7.

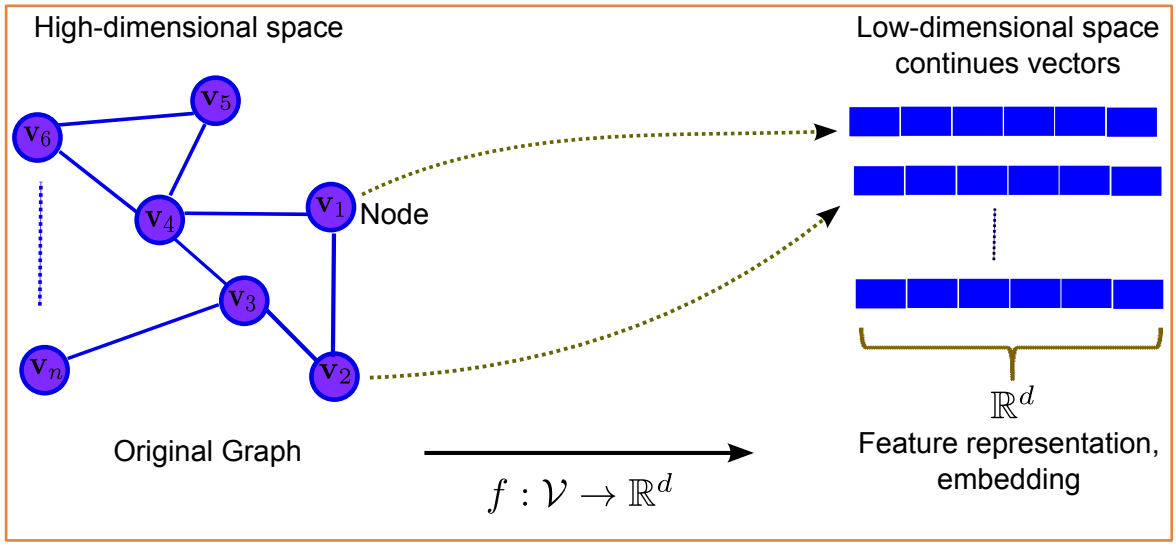

Figure 1.7: Schematic diagram of node embedding.

1.5.8 Graph Convolutional Networks

Graph neural networks have recently become the method of choice for learning from graph-structured data, which is a ubiquitous data structure extensively used in a plethora of real-world applications such as social networks and brain connectivity networks. Existing graph neural networks fall into two main categories: spatial methods and spectral methods. The former define graph convolution in the node domain as a weighted average function over neighboring nodes similar to the idea of convolution in traditional convolutional neural networks. The latter define graph convolution in the graph Fourier domain using the eigenvectors of the graph Laplacian matrix.

Spatial Methods. Convolution in spatial-based graph neural networks is defined directly in the node domain based on a node's spatial relations (i.e. neighbors). The main idea is that a node representation is updated by aggregating information from its neighboring nodes. The spatial graph convolutional operation essentially propagates node information along edges. The information that passes between neighbors and the central node in the graph is referred to as messages. During each message-passing iteration in a graph neural network, the embedding $\mathbf{h}_i^{(\ell)}$ of node $i \in \mathcal{V}$ is updated according to information aggregated from its graph neighborhood $\mathcal{N}_i$ using two neural networks AGGREGATE and UPDATE as follows:

$$\mathbf{m}_{\mathcal{N}_i}^{(\ell)} = \text{AGGREGATE}^{(\ell)}(\{\mathbf{h}_j^{(\ell)} : \ j \in \mathcal{N}_i\}). \tag{1.10}$$

and

$$\mathbf{h}_i^{(\ell+1)} = \text{UPDATE}^{(\ell)}(\mathbf{h}_i^{(\ell)}, \mathbf{m}_{\mathcal{N}_i}^{(\ell)}), \tag{1.11}$$

where $\mathbf{m}_{\mathcal{N}_i}^{(\ell)}$ is the message aggregated from the neighborhood of node i, and $\mathbf{h}_i^{(\ell+1)}$ is the embedding of node i at the $(\ell+1)$-layer.

Spectral Methods. Spectral techniques define graph convolution using graph signal processing.

- **Graph Fourier Transform:** We can generalize a convolutional network for a spectral network via graph Fourier transform based on the graph Laplacian matrix [21]. Suppose an input vector $\mathbf{x} \in \mathbb{R}^N$ is a signal defined on a graph $\mathbb{G}$ with N nodes. If $\mathbf{A} \in \mathbb{R}^{N \times N}$ is the adjacency matrix associated with a graph $\mathbb{G}$ and $\mathbf{D}$ is the diagonal degree matrix, then the normalized graph Laplacian matrix is defined as $\mathbf{L} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$. The normalized Laplacian $\mathbf{L}$ admits an eigendecomposition given by $\mathbf{L} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^\intercal$, where $\mathbf{U} = (\mathbf{u}_1, \ldots, \mathbf{u}_N)$ is an orthonormal matrix whose columns constitute an orthonormal basis of eigenvectors and $\mathbf{\Lambda} = \text{diag}(\lambda_1, \ldots, \lambda_N)$ is a diagonal matrix comprised of the corresponding eigenvalues such that $0 = \lambda_1 \leq \ldots \leq \lambda_N \leq 2$ in ascending order [41]. The graph Fourier transform of a signal $\mathbf{x} \in \mathbb{R}^N$ is defined as $\mathcal{F}(\mathbf{x}) = \hat{\mathbf{x}} = \mathbf{U}^\intercal\mathbf{x} \in \mathbb{R}^N$, and its inverse is given by $\mathcal{F}^{-1}(\hat{\mathbf{x}}) = \mathbf{U}\hat{\mathbf{x}}$.

- **Spectral Filtering of Graph Signals:** The convolution of a graph filter $\mathbf{g}$ and a graph signal $\mathbf{x}$ is defined as

$$\mathbf{g} * \mathbf{x} = \mathcal{F}^{-1}(\mathcal{F}(\mathbf{g}) \odot \mathcal{F}(\mathbf{x})) = \mathbf{U}(\mathbf{U}^\intercal\mathbf{g} \odot \mathbf{U}^\intercal\mathbf{x}), \tag{1.12}$$

  where $\odot$ denotes element-wise multiplication. Hence, applying a spectral graph filter $\mathbf{g}_{\boldsymbol{\theta}}$ on a graph signal $\mathbf{x}$ yields

$$\mathbf{g}_{\boldsymbol{\theta}}(\mathbf{L})\mathbf{x} = \mathbf{g}_{\boldsymbol{\theta}}(\mathbf{U}\mathbf{\Lambda}\mathbf{U}^\intercal)\mathbf{x} = \mathbf{U}\mathbf{g}_{\boldsymbol{\theta}}(\mathbf{\Lambda})\mathbf{U}^\intercal\mathbf{x}, \tag{1.13}$$

  where $\boldsymbol{\theta}$ is a vector of learnable parameters. However, there are three limitations that prohibit the spectral filter from being used in practice: the filter is not localized, the learning complexity is $\mathcal{O}(N^2)$ due to matrix-vector multiplication, and the number of parameters depends on the input size. To tackle these limitations, the spectral filter can be approximated using Chebyshev polynomials as follows

$$\mathbf{g}_{\boldsymbol{\theta}}(\mathbf{\Lambda}) = \sum_{k=0}^{K-1} \theta_k T_k(\hat{\mathbf{\Lambda}}), \tag{1.14}$$

  where the Chebyshev polynomials are defined recursively by

$$T_k(x) = 2xT_{k-1}(x) - T_{k-2}(x) \quad \text{with } T_0 = 1 \text{ and } T_1 = x \tag{1.15}$$

  and $\hat{\mathbf{\Lambda}} = 2\mathbf{\Lambda}/\lambda_{\text{max}} - \mathbf{I}$ is a diagonal matrix of scaled eigenvalues with λ_{max} denoting the largest eigenvalue of the Laplacian matrix. Hence, the cost of the resulting filtering operation is reduced to $\mathcal{O}(K|\mathcal{E}|)$.

1.5.9 Brain Connectivity Networks

The human brain is a network comprised of spatially distributed, but functionally linked regions that continuously share information with each other. The goal of brain network analysis is to predict human brain diseases such as neurodegenerative diseases, which cause the brain and nerves to deteriorate over time. Unfortunately, there is no cure for neurodegenerative diseases, but treatment can still help. Brain networks can be modeled as graphs consisting of nodes connected by edges. Brain connectivity can be categorized into three main types [42]:

- **Anatomical Connectivity:** Structural connectivity or anatomical connectivity refers to a network of physical or structural connections linking sets of neurons or neuronal elements. In the graph theoretical analysis of anatomical brain connectivity, the white matter connections between regions of the brain are identified and serve as basis for the assessment of regional connectivity profiles.

- **Functional Connectivity:** Functional connectivity provides statistical dependencies of neuronal activation patterns of different brain regions. In other words, two regions are considered to show functional connectivity if there is a statistical relationship between the measures of activity recorded for them.

- **Effective Connectivity:** Effective connectivity describes the causal influences that neural units exert over another. While functional connectivity only describes statistical dependencies between spatially segregated neuronal events, effective connectivity refers to causal interactions.

Brain network analysis leverage graph theory to represent the brain as a graph consisting of a set of nodes connected by edges or links. As shown in Figure 1, the brain network construction pipeline is comprised of four main steps: computing a brain atlas, defining brain regions of interest (ROIs), extracting the timeseries associated with ROIs, and finally estimating the functional connectivity metrics from these timeseries [42].

- **Predefined Atlases:** An atlas generally accounts for a certain state of the knowledge of the brain structures (anatomical or functional) from which well-defined entities can be distinguished. In other words, an atlas represents a certain labeling of brain structures. Applying a predefined brain atlas from a single brain is not be ideal for region-based analysis due to anatomical variability.

- **Region of Interests of Brain Images:** The main goal of region of interests of brain images is to explore the underlying signal behind brain analysis when investigating a region for effects.
- **Timeseries Signals Extraction:** Timeseries are extracted from each ROI for each subject and then used to measure the functional connectivity between pairs of nodes in graph. One of the techniques to compute these quantities is the average of Haemodynamic response function of the fMRI time series signals over all voxels for each region [43].
- **Functional Connectivity Estimation:** Functional connectivity can be estimated via the covariance between signals from brain ROIs from different sensors/time spots.
- **Brain Functional Parcellation:** ROIs can be extracted from fMRI via two methods: 1) clustering approaches such as K-means algorithm, Ward's algorithm, spectral clustering and geometric clustering; and 2) dictionary learning techniques such as independent components analysis and variants of principal components analysis [42].

Figure 1.8 shows the steps of labeled graph construction. The first step is to compute the brain atlas directly from the data, followed by extracting brain regions from the computed atlas. The second step is to compute time series, and then find the connectivity matrix by using correlation between time series (i.e. correlation, partial correlation, or covariance). The last step is to construct a brain graph (connectome) to be used as input to the deep neural network, where the nodes of the graph are represented by regions of interest and the connectivity between edges express the links (edges) of the graph.

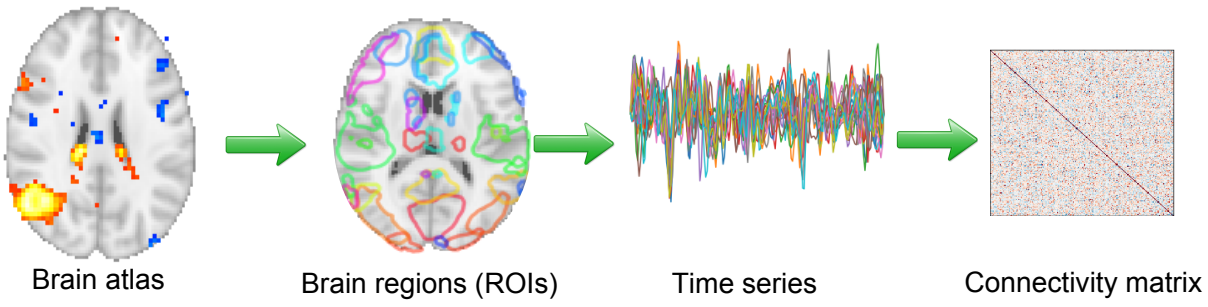

Figure 1.8: Functional connectivity construction.

1.6 Overview and Contributions

The organization of this thesis is as follows:

- In Chapter 1, we begin with the basic concepts which we refer to throughout the thesis. Then, we present the motivations and the goals of this research, followed by the problem statement, the objective of this study, and literature review. We also present an overview of triangular mesh representation, regression analysis, convolutional neural networks, medical image segmentation, graph representation learning, and brain connectivity networks.

- In Chapter 2, we introduce a geometric feature-preserving framework for carpal bone surface denoising using graph signal processing. The proposed approach is formulated as a constrained optimization problem with an objective function consisting of a fidelity term specified by a noise model and a regularization term associated with data prior. These terms are weighted by a normalized mesh Laplacian, which is defined in terms of a data-adaptive kernel similarity matrix in conjunction with a matrix balancing procedure. Minimizing such an objective function is carried out by iteratively solving a sparse system of linear equations using the conjugate gradient method.

- In Chapter 3, we propose a two-stage RidgeNet model for bone age assessment using instance segmentation and ridge regression. In the first stage, we employ an image annotation and segmentation model to annotate and segment the hand from the radiographic image, followed by background removal. In the second stage, we design a regression neural network architecture comprised of a pre-trained convolutional neural network for learning salient features from the segmented pediatric hand radiographs and a ridge regression output layer for predicting the pediatric bone age.

- In chapter 4, we develop an aggregator normalization graph convolutional network by exploiting not only aggregation in graph sampling, but also skip connections and identity mapping. The proposed graph neural network learns discriminative graph node representations by incorporating both imaging and non-imaging features into the graph nodes and edges, respectively, with the aim of boosting predictive capabilities.

- In Chapter 5, we present a summary of the contributions of this thesis and limitations, and we also outline several directions for future research in this area of study.

CHAPTER 2

Carpal Bone Surface Denoising

We present a geometric framework for surface denoising using graph signal processing, which is an emerging field that aims to develop new tools for processing and analyzing graph-structured data. The proposed approach is formulated as a constrained optimization problem whose objective function consists of a fidelity term specified by a noise model and a regularization term associated with data prior. Both terms are weighted by a normalized mesh Laplacian, which is defined in terms of a data-adaptive kernel similarity matrix in conjunction with matrix balancing. Minimizing the objective function reduces to iteratively solving a sparse system of linear equations via the conjugate gradient method. Extensive experiments on noisy carpal bone surfaces demonstrate the effectiveness of our approach in comparison with existing methods. We perform both qualitative and quantitative comparisons using various evaluation metrics.

2.1 Introduction

Recent advances in 3D scanning technology have led to an increasing use of 3D models in many fields, including the entertainment industry, archaeology, computer vision, and medical imaging. These models are usually captured in the form of point clouds or polygonal meshes [44], but they are often corrupted by noise during the data acquisition stage. The main problem with 3D shape denoising is how we can distinguish between noise and features, especially sharp surface features. To ensure high-quality 3D shapes for use in downstream applications, it is important to develop effective surface denoising techniques to remove inevitable noise in the measurements [7, 10, 11, 45–48].

In recent years, a plethora of techniques have been proposed to tackle the 3D surface denoising problem. Generally, surface denoising methods may be classified into two major categories: isotropic and anisotropic. The former techniques filter the noisy data independently of direction, while the latter methods modify the diffusion equation to make it nonlinear or anisotropic in order to preserve the sharp features of a 3D mesh surface. The simplest surface denoising method is the Laplacian flow, which repeatedly and simultaneously adjusts the location of each mesh vertex to the geometric center of its neighboring vertices [7].

Most surface denoising methods are adopted from the image processing literature [1, 49–51], including the mean, median, and bilateral filters. In particular, bilateral filtering has been used extensively in image processing applications, due in large part to its good performance in smoothing noisy images while preserving edges. The bilateral filter takes into account the variation of image intensities by replacing the intensity value at a pixel by a weighted average of the intensity values from neighboring pixels. Although these filters have been successfully applied to image denoising, it is, however, not straightforward to apply them directly to graph-structured data. Fleishman *et al.* [10] proposed a bilateral mesh denoising approach that filters each mesh vertex in the normal direction using local neighborhoods. Zheng *et al.* [11] applied the bilateral normal filter in a local iterative and a global non-iterative scheme for anisotropic denoising. Sun *et al.* [52] introduced a two-step mesh denoising framework. In the first step, the noisy face normals are filtered iteratively by weighted averaging of neighboring face normals. In the second step, the mesh vertex positions are iteratively updated based on the denoised face normals. Huang and Uscher proposed a multiscale anisotropic Laplacian (MSAL) [53], which employs the anisotropic Laplacian operator combined with a roughness scale, and yields significantly better results than the anisotropic Laplacian and the bilateral filter. Ouafdi *et al.* [54] introduced a probabilistic mesh denoising method by performing an anisotropic average of neighboring vertices weighted by a Riemannian metric. Zhang *et al.* [12] presented a guided mesh normal filtering framework by constructing the guidance for joint bilateral filtering of geometry signals using a two-step process. Joint bilateral filtering is applied to the face normals, followed by updating the mesh vertices to agree with the denoised face normals. More recently, Yadav *et al.* [55] proposed a two-stage mesh denoising approach using robust statistics. In the first stage, the face normals are filtered via bilateral normal filtering using Tukey's bi-weight as a similarity function. In the second stage, the mesh vertex positions are updated using edge-to-face normal orthogonality constraints along with differential coordinates.

On the other hand, image/surface denoising via graph signal processing techniques has received considerable attention in recent years [1, 56, 57]. A graph-based approach to image denoising and deblurring was introduced in [1] using a data-adaptive objective function derived from a normal-

ized graph Laplacian. Chung *et al.* [57] used the graph Laplacian to construct the discrete version of heat kernel smoothing on graph-structured data, obtained by binary segmentation of the computed tomography of human lung data. Also, Chung *et al.* [58] introduced a heat kernel regression approach to surface smoothing using the Laplace-Beltrami eigenfunctions, which are obtained by solving a generalized eigenvalue problem. Such an approach can, however, be prohibitively expensive, especially when the problem size is large (i.e. large matrices). Another issue with spectral approaches is how to select the appropriate number of eigenvalues and associated eigenfunctions to be retained.

Motivated by the good performance of the similarity-based image denoising framework proposed in [1], we introduce a simple, yet effective feature-preserving approach to 3D mesh denoising. The proposed method employs a normalized mesh Laplacian, which is defined in terms of a data-adaptive kernel similarity matrix in conjunction with matrix balancing. We formulate our surface denoising framework as a constrained minimization problem, which can be solved efficiently using the conjugate gradient (CG) method. Our approach can remove the noise effectively while preserving the nonlinear features of surfaces, such as curved surface regions, sharp edges, and fine details. While our proposed framework is general enough to be applied to any problem involving surface denoising, the primary focus of this work is on noise removal from carpal bone surfaces. Further, recovering high quality surfaces from noisy carpal bone surfaces is a fundamental problem in computational anatomy and biomechanics, and is of paramount importance to the diagnosis of wrist pathologies, such as arthritis. Our main contributions may be summarized as follows:

- We introduce a mesh denoising approach using a data-adaptive kernel similarity matrix in conjunction with matrix balancing.
- We formulate the proposed framework as a constrained minimization problem, and solve it iteratively using the conjugate gradient method.
- Our experimental results show superior performance of the proposed framework over existing mesh denoising methods.

The rest of this chapter is organized as follows. In Section 2, we briefly recall some basic concepts of geometry processing, followed by a general formulation of the surface denoising problem in the graph signal processing setting. In Section 3, we present the main building blocks of our method, and discuss in detail the algorithmic steps. In Section 4, we present experimental results to demonstrate the competitive performance of our denoising approach on carpal bone surfaces. Finally, Section 5 concludes the chapter and points out future work directions.

2.2 Problem Formulation

Triangular mesh representation: A 3D shape is usually modeled as a triangle mesh $\mathbb{M}$ whose vertices are sampled from a Riemannian manifold. A triangle mesh $\mathbb{M}$ may be defined as a graph $\mathbb{G} = (\mathcal{V}, \mathcal{E})$ or $\mathbb{G} = (\mathcal{V}, \mathcal{T})$, where $\mathcal{V} = \{\mathbf{v}_1, \ldots, \mathbf{v}_n\}$ is the set of vertices, $\mathcal{E} = \{\mathbf{e}_{ij}\}$ is the set of edges, and $\mathcal{T} = \{\mathbf{t}_1, \ldots, \mathbf{t}_m\}$ is the set of triangles. Each edge $\mathbf{e}_{ij} = [\mathbf{v}_i, \mathbf{v}_j]$ connects a pair of vertices $\{\mathbf{v}_i, \mathbf{v}_j\}$. Two distinct vertices $\mathbf{v}_i, \mathbf{v}_j \in \mathcal{V}$ are adjacent (denoted by $\mathbf{v}_i \sim \mathbf{v}_j$ or simply $i \sim j$) if they are connected by an edge, i.e. $\mathbf{e}_{ij} \in \mathcal{E}$. The neighborhood of a vertex $\mathbf{v}_i$ is the set $\mathring{\mathbf{v}}_i = \{\mathbf{v}_j \in \mathcal{V} : \mathbf{v}_j \sim \mathbf{v}_i\}$.

Laplacian matrix of a weighted graph: The graph $\mathbb{G}$ may be equipped with a nonnegative weight function $\omega : \mathcal{V} \times \mathcal{V} \to \mathbb{R}^+$ such that

$$\omega_{ij} = \begin{cases} \omega_{ii} & \text{if } \ i = j \\ \omega_{ij} & \text{if } \ i \sim j \\ 0 & \text{o.w.} \end{cases} \tag{2.1}$$

The Laplacian matrix $\mathbf{L} = (\ell_{ij})$ of a weighted graph is defined as $\mathbf{L} = \mathbf{D} - \mathbf{A}$, whose elements are given by

$$\ell_{ij} = \begin{cases} d_i - \omega_{ii} & \text{if } \ i = j \\ -\omega_{ij} & \text{if } \ i \sim j \\ 0 & \text{o.w.} \end{cases} \tag{2.2}$$

where $\mathbf{A} = (w_{ij})$ is the weighted adjacency matrix, and $\mathbf{D} = \operatorname{diag}(d_1, \ldots, d_n)$ is the degree matrix with $d_i = \sum_{j \sim i} \omega_{ij}$, the degree of vertex i. The normalized weighted Laplacian matrix $\boldsymbol{\mathcal{L}}$ is defined as

$$\boldsymbol{\mathcal{L}} = \mathbf{D}^{-1/2}\mathbf{L}\mathbf{D}^{-1/2} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}. \tag{2.3}$$

Figure 2.1 displays a 3D hand model and its weighted Laplacian matrix, with weights $\omega_{ij} = \|\mathbf{v}_i - \mathbf{v}_j\|$, where $\|\cdot\|$ denotes the Euclidean norm. The sparsity pattern (or support) of $\mathbf{L} = (\ell_{ij})$ is the set of indices ij with $\ell_{ij} \neq 0$.

The Laplacian matrix may be viewed as an operator defined on the space of graph signals $u : \mathcal{V} \to \mathbb{R}$ as follows:

$$\mathbf{L}u(i) = \sum_{i \sim j} \omega_{ij}(u(i) - u(j)), \quad \text{for all } i \in \mathcal{V}. \tag{2.4}$$

In other words, $\mathbf{L}u(i)$ is the sum of the weighted differences between the value of the graph signal u at vertex i and the values at the neighboring vertices.

Since $|\mathcal{V}| = n$, we may represent any graph signal $u : \mathcal{V} \to \mathbb{R}$ as a column vector $\mathbf{u} = (u(i)) \in \mathbb{R}^n$ with the ith element $u(i)$. Thus, the quadratic form of the signal $\mathbf{u}$ with respect to the Laplacian

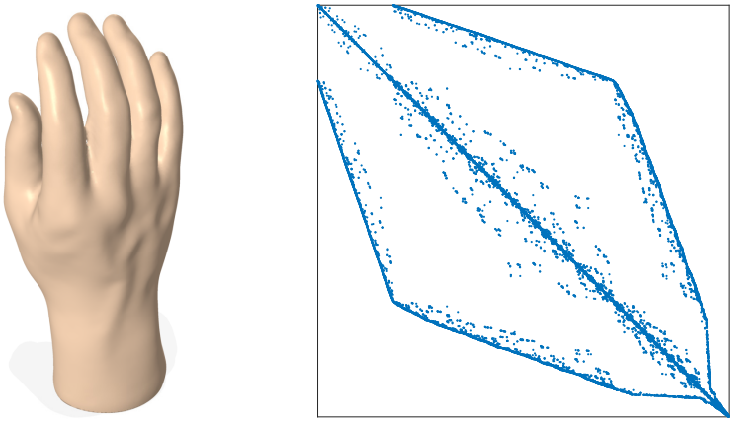

Figure 2.1: Hand model (left) and sparsity pattern plot of its weighted Laplacian matrix (right).

matrix can be expressed as

$$\mathbf{u}^{\mathsf{T}}\mathbf{L}\mathbf{u} = \sum_{i \sim j} \omega_{ij}(u(i) - u(j))^2, \tag{2.5}$$

which shows that if the weights are symmetric, then the Laplacian matrix is symmetric, positive semi-definite. So the action of the Laplacian on a signal may be viewed as measuring the smoothness of that signal across the edges in the mesh.

2.2.1 Mesh Denoising Model

In all real applications, measurements are usually perturbed by noise. In the course of acquiring, transmitting or processing a 3D model for example, the noise-induced degradation often yields a resulting graph signal observation model, and the most commonly used is the additive one,

$$\mathbf{v} = \mathbf{u} + \boldsymbol{\eta}, \tag{2.6}$$

where the observed graph signal $\mathbf{v}$ includes the original graph signal $\mathbf{u}$ and the random noise process $\boldsymbol{\eta}$, which is usually assumed to be Gaussian with zero mean and standard deviation σ.

Surface denoising refers to the process of recovering a 3D model contaminated by noise. The challenge of the problem of interest lies in recovering the graph signal $\mathbf{u}$ from the observed signal $\mathbf{v}$, and furthering the estimation by making use of any prior knowledge/assumptions about the noise process $\boldsymbol{\eta}$.

When considering the noise model (2.6), our goal may be succinctly stated as one of estimating the underlying graph signal $\mathbf{u}$ based on an observed signal $\mathbf{v}$ and/or any potential knowledge of the noise statistics to further regularize the solution. This yields the following fidelity-constrained

optimization problem

$$\begin{aligned} \min_{\mathbf{u}} \quad & \mathcal{R}(\mathbf{u}) \\ \text{s.t.} \quad & \|\mathbf{v}-\mathbf{u}\|^2 \leq \sigma^2 \end{aligned} \tag{2.7}$$

where $\mathcal{R}$ is a given regularization functional, which often defines the particular emphasis on the features of the achievable solution. In other words, we want to find an optimal solution that yields the smallest value of the objective function among all solutions that satisfy the constraints. Using Lagrange's theorem, the minimizer of (2.7) is given by

$$\hat{u} = \arg\min_{\mathbf{u}} \left\{ \|\mathbf{v}-\mathbf{u}\|^2 + \beta \mathcal{R}(\mathbf{u}) \right\}, \tag{2.8}$$

where β is a nonnegative regularization parameter, which is often estimated or chosen *a priori*. A critical issue, however, is the choice of the regularization functional $\mathcal{R}$, which is often driven by geometric arguments. A commonly used functional is the mesh Laplacian quadratic form defined as a (squared) weighted vector norm

$$\mathcal{R}(\mathbf{u}) = \|\mathbf{u}\|_{\mathbf{L}}^2 = \mathbf{u}^{\mathsf{T}}\mathbf{L}\mathbf{u}. \tag{2.9}$$

2.3 Method

In this section, we present the main components of the proposed surface denoising framework, and describe in detail its algorithmic steps. The flowchart of our approach is illustrated in Figure 2.2.

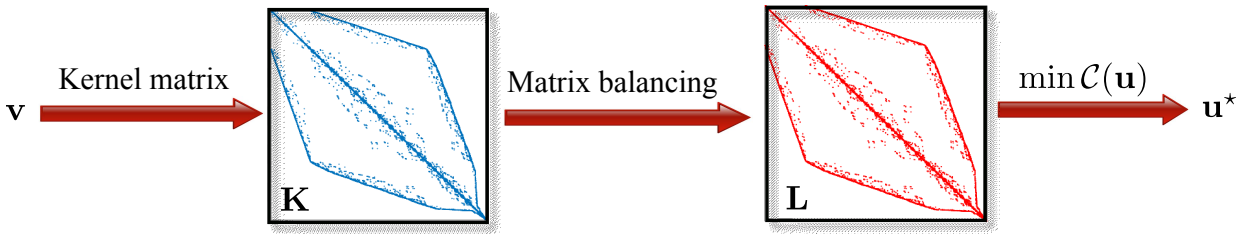

Figure 2.2: Flowchart of our proposed surface denoising method, where $\mathbf{v}$ is the noisy graph signal and $\mathbf{u}^\star$ is the estimated signal.

Kernel similarity: Using the Gaussian kernel, we define the kernel weight matrix $\mathbf{S} = (s_{ij})$ as

$$s_{ij} = \exp\left(-\frac{\|\mathbf{v}_i - \mathbf{v}_j\|^2}{2h^2}\right), \tag{2.10}$$

where $\mathbf{v}_i$ is the ith vertex of the noisy mesh, $\mathbf{v}_j$ are the neighboring vertices of $\mathbf{v}_i$, and h is the bandwidth parameter of the Gaussian kernel. Each edge weight s_{ij} is a similarity measure whose

value is large when i is closer to j. We define the kernel similarity weight matrix as follows:

$$\mathbf{K} = \frac{\mathbf{S} + \mathbf{S}^\intercal}{2}, \tag{2.11}$$

which is a symmetric, non-negative matrix. Further, all of its off-diagonal elements are positive.

Sinkhorn matrix balancing: Applying the Sinkhorn matrix balancing procedure [59] to the kernel similarity weight matrix $\mathbf{K}$ yields a symmetric non-negative doubly stochastic filtering matrix $\mathbf{W}$ given by

$$\mathbf{W} = \mathbf{C}^{-1/2}\mathbf{K}\mathbf{C}^{-1/2}, \tag{2.12}$$

where $\mathbf{C}$ is a diagonal scaling matrix [60]. It should be noted that since the filtering matrix $\mathbf{W}$ is doubly stochastic, its largest eigenvalue is equal to 1 with associated eigenvector $\mathbf{e} = \mathbf{1}/\sqrt{n}$, where $\mathbf{1}$ is a vector of all ones. In other words, the filtering matrix preserves the DC component of a graph signal (i.e. $\mathbf{W}\mathbf{e} = \mathbf{e}$).

Normalized mesh Laplacian: We define the normalized mesh Laplacian matrix as

$$\mathbf{L} = \mathbf{I} - \mathbf{W} = \mathbf{I} - \mathbf{C}^{-1/2}\mathbf{K}\mathbf{C}^{-1/2}, \tag{2.13}$$

which is symmetric, positive semi-definite. The Laplacian matrix $\mathbf{L}$ can be interpreted as a data-adaptive high-pass filter, enabling us to incorporate a variety of filters in the data term as well the regularization term.

From (2.13), it is easy to see that if λ is an eigenvalue of $\mathbf{W}$, then $1 - \lambda$ is an eigenvalue of $\mathbf{L}$. In particular, 0 is an eigenvalue of $\mathbf{L}$ with associated eigenvector $\mathbf{e}$. The eigenvalues of $\mathbf{L}$ may be viewed as graph frequencies. Moreover, the eigenvectors associated with the smallest eigenvalues have smooth oscillations and capture well the large-scale properties of a shape. As shown in Figure 2.3, the (non-trivial) eigenvectors of $\mathbf{L}$ encode important information about the global geometry of a shape. Notice that the eigenvectors associated with larger eigenvalues oscillate more rapidly. Blue regions indicate small values of the eigenvectors and red colors regions indicate large values, while green and yellow regions in between.

2.3.1 Surface Denoising Approach

We formulate our surface denoising framework as a constrained optimization problem by minimizing the following cost function

$$\begin{aligned} \mathcal{C}(\mathbf{u}) &= \|\mathbf{v} - \mathbf{u}\|^2_{\mathbf{I}+\alpha\mathbf{L}} + \beta\|\mathbf{u}\|^2_{\mathbf{L}} \\ &= (\mathbf{v} - \mathbf{u})^\intercal(\mathbf{I} + \alpha\mathbf{L})(\mathbf{v} - \mathbf{u}) + \beta\mathbf{u}^\intercal\mathbf{L}\mathbf{u}, \end{aligned} \tag{2.14}$$

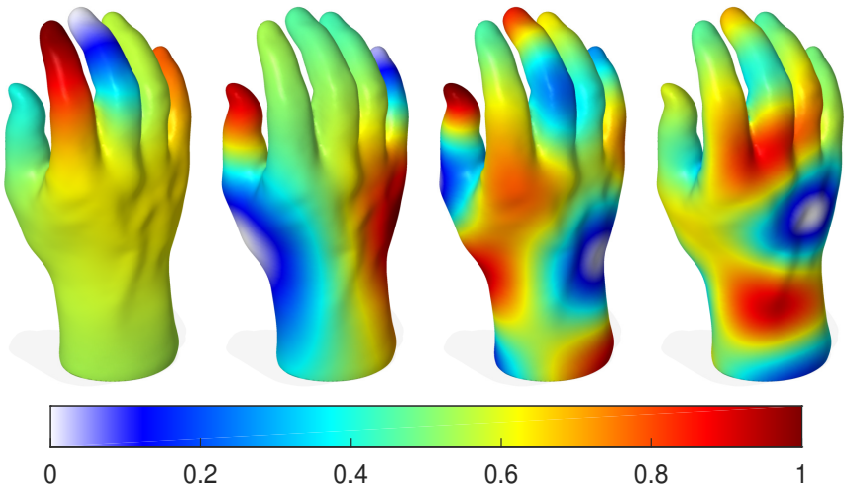

Figure 2.3: Visualization of four eigenvectors of the normalized mesh Laplacian matrix. From left to right: a 3D hand model Gouraud shaded and color-coded by the values of the second, eighth, fifteenth and twentieth eigenvectors.

where $\mathbf{v}$ is the noisy graph signal and $\mathbf{u}$ is the estimated signal. The non-negative parameters α and β are often estimated or chosen *a priori*. Note that the first term is a weighted error between the input and its estimate, and minimizing such an error yields a solution as close as possible to the input. Minimizing the second term, on the other hand, yields a smooth solution. Further, $\mathbf{I} + \alpha\mathbf{L}$ is a symmetric, positive-definite matrix.

The cost function $\mathcal{C}(\mathbf{u})$ can be minimized by finding its gradient and setting it to zero

$$\nabla\mathcal{C}(\mathbf{u}) = -2(\mathbf{I} + \alpha\mathbf{L})(\mathbf{v} - \mathbf{u}) + 2\beta\mathbf{L}\mathbf{u} = \mathbf{0}, \tag{2.15}$$

resulting in the following system of linear equations

$$\big(\mathbf{I} + (\alpha + \beta)\mathbf{L}\big)\mathbf{u} = \big(\mathbf{I} + \alpha\mathbf{L}\big)\mathbf{v}. \tag{2.16}$$

Since $\mathbf{I} + (\alpha + \beta)\mathbf{L}$ is a symmetric, positive-definite matrix, the system (2.16) can be efficiently solved using iterative methods such as the CG method, which is a commonly used iterative algorithm for solving sparse systems of linear equations.

2.3.2 Algorithm

The objective of 3D mesh denoising is to remove noise while preserving features. Our proposed surface denoising approach consists of two major steps, as illustrated in Figure 2.2. In the first step, we compute the normalized mesh Laplacian using kernel similarity and matrix balancing. In the second step, we iteratively solve a sparse system of linear equations using the CG method. It should be noted that the proposed algorithm consists of both outer and inner iterations. The

outer iterative process is used to compute the normalized mesh Laplacian, while the inner iterative process is employed to solve the constrained minimization problem. Algorithm 2 summarizes the main algorithmic steps of our approach.

Algorithm 2 Feature-Preserving Mesh Denoising

Input Noisy graph signal $\mathbf{v}$

$\hat{\mathbf{u}}^{(0)} = \mathbf{v}$
$k = 0$.
while not converged **do**
    Compute the kernel similarity weight matrix $\mathbf{K}$ from $\hat{\mathbf{u}}^{(k)}$ using (2.10)-(2.11)
    Apply Sinkhorn matrix balancing to $\mathbf{K}$ to get the diagonal matrix $\mathbf{C}$
    Compute the Laplacian matrix $\mathbf{L} = \mathbf{I} - \mathbf{C}^{-1/2}\mathbf{K}\mathbf{C}^{-1/2}$
    Solve the linear system in (2.16) using conjugate gradient to estimate $\hat{\mathbf{u}}^{(k+1)}$.
    Set $\mathbf{u}^{\star} = \hat{\mathbf{u}}^{(k+1)}$
    $k = k + 1$
end while

return $\mathbf{u}^{\star}$

Output Estimated signal $\mathbf{u}^{\star}$

2.4 Experiments

In this section, we evaluate through extensive experiments the performance of our proposed mesh denoising approach on carpal bone surfaces [61]. As shown in Figure 2.4, the carpal bones of the right wrist in a healthy male are the capitate, hamate, lunate, pisiform, scaphoid, trapezium, trapezoid, triquetrum. Since the trapeziometacarpal joint of the thumb is a common site of osteoarthritis, the first metacarpal bone is also considered in our analysis. The forearm's radius and ulna bones, which support the many muscles that manipulate the bones of the hand and wrist, are also depicted in Figure 2.4.

Implementation details: All experiments were performed on a desktop computer with an Intel Core 2 Duo running at 3.40 GHz and 16 GB RAM; and the proposed mesh denoising algorithm was implemented in MATLAB. The parameters α and β are chosen as the inverse of the minimum and maximum of the mesh degree values, respectively (i.e. $\alpha = 1/d_{min}$ and $\beta = 1/d_{max}$). The kernel bandwidth parameter h is estimated using the median absolute deviation (MAD) as follows:

$$h = 1.4826 \sum_{i=1}^{n} \underset{j \sim i}{\text{MAD}}(\mathbf{v}_i - \mathbf{v}_j), \tag{2.17}$$

where $\text{MAD}(\mathbf{x}) = \text{median}(\|\mathbf{x} - \text{median}(\|\mathbf{x}\|)\|)$ is a measure of spread that represents expected absolute-error loss, and is robust to outliers.

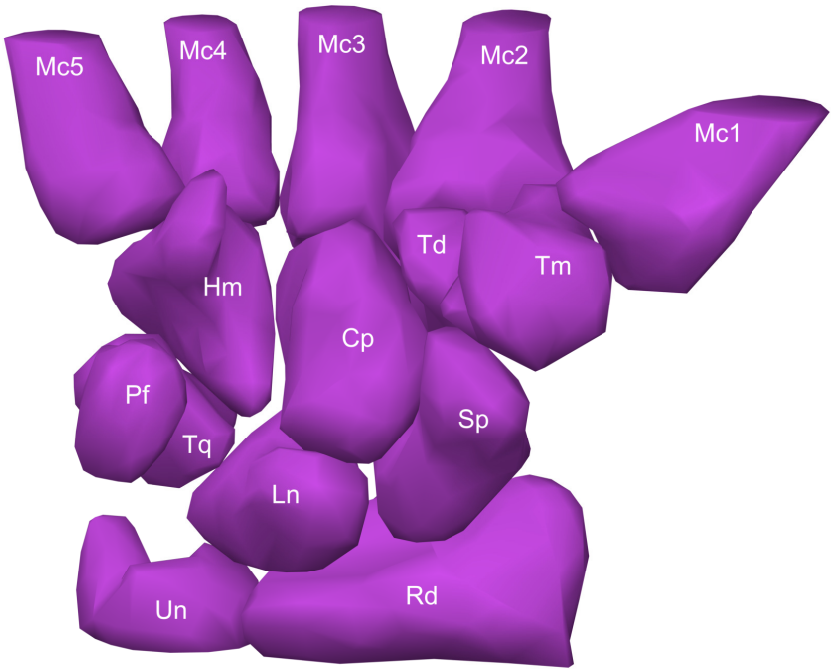

Figure 2.4: Carpal bone anatomy of a healthy male from a palmar view. The carpus consists of eight carpal bones, which are arranged in proximal and distal rows. The proximal row contains scaphoid (Sp), lunate (Ln), triquetrum (Tq) and pisiform (Pf), while the distal row contains trapezium (Tm), trapezoid (Td), capitate (Cp), and hamate (Hm). The distal row adjoins the five metacarpals (Mc1-5) of the wrist. The radius (Rd) and ulna (Un) are also shown.

Baseline methods: We compare the effectiveness of our proposed technique with several state-of-the-art approaches, including bilateral mesh denoising (BMD) [10], multiscale anisotropic Laplacian (MSAL) method [53], guided mesh normal denoising (GMD) [12], and robust and high fidelity mesh denoising (RMD) [55].

2.4.1 Results

We performed extensive experiments on various carpal bone surfaces, including right metacarpal, scaphoid, left metacarpal, left hamate, lunate and pisiform, as shown in Figure 2.5.

We generate the noisy carpal bones models by setting the standard deviation of the noise to $0.5\bar{\ell}$ and $0.7\bar{\ell}$ of the mean edge length $\bar{\ell}$ given by

$$\bar{\ell} = \frac{1}{|\mathcal{E}|} \sum_{\mathbf{e}_{ij} \in \mathcal{E}} \|\mathbf{e}_{ij}\|, \tag{2.18}$$

where $\|\mathbf{e}_{ij}\| = \|\mathbf{v}_i - \mathbf{v}_j\|$ if $i \sim j$, and $\|\mathbf{e}_{ij}\| = 0$ otherwise. More precisely, a vertex $\mathbf{v}_i$ of a noisy mesh is given by the additive random noise model:

$$\mathbf{v}_i = \mathbf{u}_i + \sigma(\boldsymbol{\eta}_i \odot \mathbf{n}_i), \tag{2.19}$$

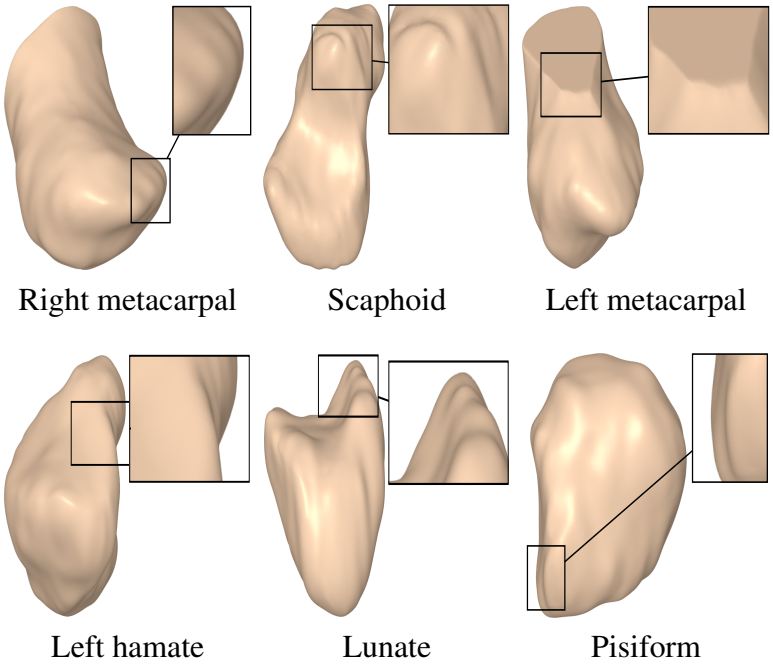

Figure 2.5: Carpal bones models.

where $\boldsymbol{\eta}_i$ are i.i.d. Gaussian random vectors (i.e. $\boldsymbol{\eta}_i$ is a 3-dimensional vector containing pseudo-random values drawn from the standard normal distribution $N(0,1)$), $\mathbf{n}_i$ is the unit normal vector at the noise-free vertex $\mathbf{u}_i$, and $\odot$ denotes the Hadamard product between two vectors (i.e. the elements of the vector $\boldsymbol{\eta}_i \odot \mathbf{n}_i$ are obtained via element-by-element multiplication of the vectors $\boldsymbol{\eta}_i$ and $\mathbf{n}_i$).

Qualitative Comparison

The visual comparison was performed with the most prevalent methods of 3D mesh denoising, including BMD [10], MSAL [53], GMD [12] and RMD [55]. In Figure 2.6, the noisy right metacarpal model is generated by adding a Gaussian noise with standard deviation $\sigma = 0.5$ to the vertices of the ground truth mesh along the vertex normals. As can be seen, the output results of BMD, MSAL, GMD and RMD still contain a considerable amount of noise in some regions of the denoised model, while the proposed approach removes well the noise and at the same time preserves the surface details. Figure 2.7 displays the denoising results on the noisy scaphoid, left metacarpal and left hamate models with a noise standard deviation $\sigma = 0.5$ proportional to the mean edge length of the mesh. Notice again that the proposed approach preserves well the edges, while RMD tends to over-smooth the features. Further, the noise is mostly eliminated using our approach removes without affecting flat regions. Further, the sharp features are well preserved, as depicted in the enlarged views, which show that the geometric structures and the fine details of the

denoised carpal bone models are very well preserved.

Figure 2.8 shows the denoising results on the noisy scaphoid, lunate and pisiform models with a higher noise standard deviation $\sigma = 0.7$ proportional to the mean edge length of the mesh. As can be seen, RMD removes relatively well the noise but does not preserve the sharp features. The other baseline methods do not remove well the noise and also tend to over-smooth the sharp regions, while our approach effectively removes the noise without creating any edge flips. While RMD yields comparable results to our approach, it does not, however, preserve edges with the same effectiveness.

In all the experiments, we observe that our approach is able to suppress noise while preserving important geometric features of the carpal bone surfaces in a fast and efficient manner. This better performance is in fact consistent with a large number of 3D models used for experimentation.

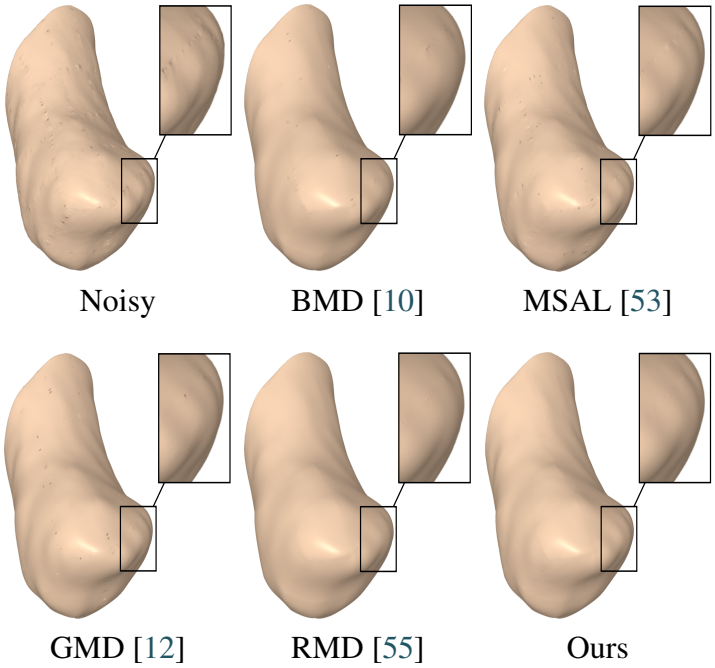

Figure 2.6: Surface denoising results on the noisy right metacarpal model corrupted by Gaussian noise with $\sigma = 0.5$. The magnified views of denoised models show that our method outperforms the baselines in preserving the surface features.

Quantitative Comparison

To quantify the difference between the ground truth and estimated model, we use three different measures, namely the mean orientation error metric, face-normal error metric, and face quality metric [55].

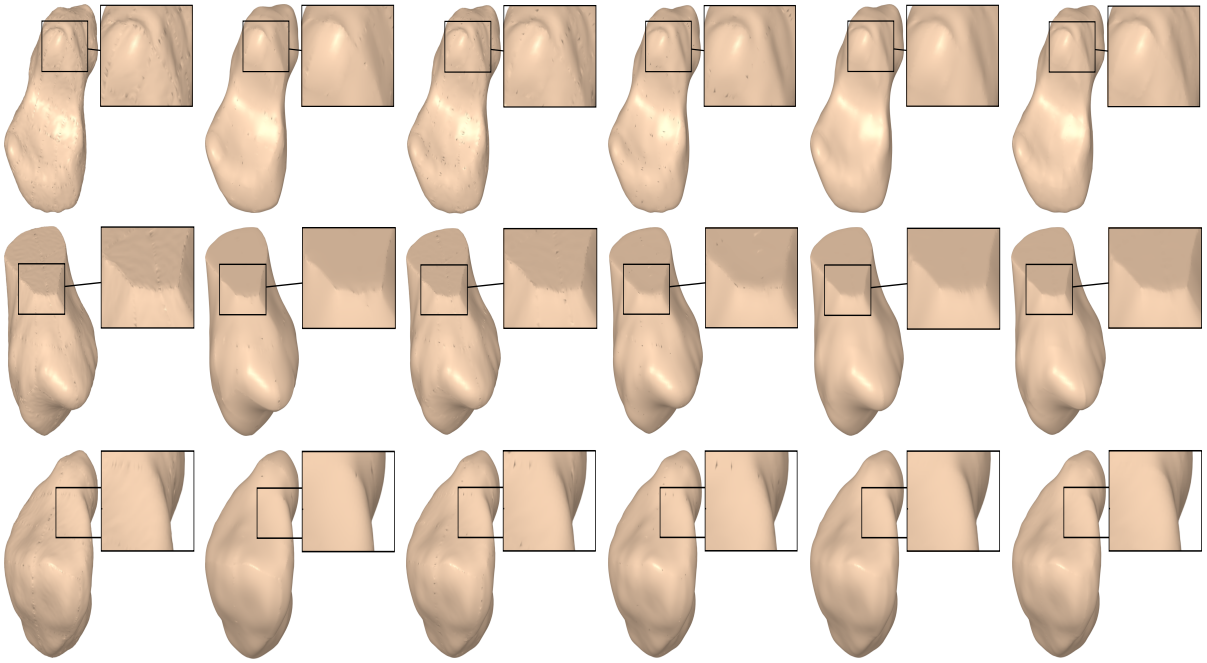

Figure 2.7: Surface denoising results on the noisy scaphoid, left metacarpal and left hamate models. The noise standard deviation is set to $\sigma = 0.5$.

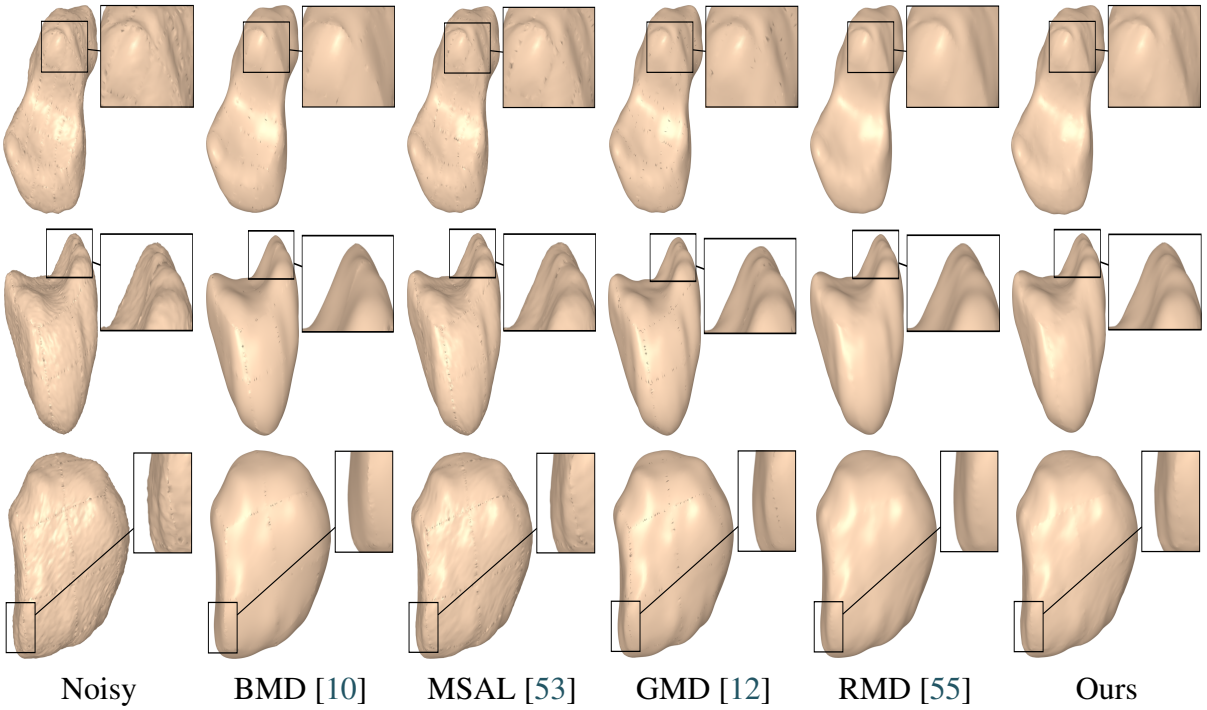

Figure 2.8: Surface denoising results on the noisy scaphoid, lunate and pisiform models. The noise standard deviation is set to $\sigma = 0.7$.

Let $\mathbb{M} = (\mathcal{V}, \mathcal{T})$ and $\widehat{\mathbb{M}} = (\widehat{\mathcal{V}}, \widehat{\mathcal{T}})$ be the original and denoised models with vertex sets $\mathcal{V} = \{\mathbf{v}_i\}_{i=1}^n$ and $\widehat{\mathcal{V}} = \{\hat{\mathbf{v}}_i\}_{i=1}^n$, and triangle sets $\mathcal{T} = \{\mathbf{t}_j\}_{j=1}^m$ and $\widehat{\mathcal{T}} = \{\hat{\mathbf{t}}_j\}_{j=1}^m$, respectively.

Mean orientation error metric: The orientation error between the original model and the denoised one can be measured using the mean orientation error metric given by

$$E_o = \frac{1}{m} \sum_{j=1}^{m} \angle(\mathbf{n}(\mathbf{t}_j), \mathbf{n}(\hat{\mathbf{t}}_j)), \tag{2.20}$$

where $\mathbf{n}(\mathbf{t}_j)$ and $\mathbf{n}(\hat{\mathbf{t}}_j)$ are the unit faces normals of $\mathbf{t}_j$ and $\hat{\mathbf{t}}_j$, respectively. The symbol $\angle$ denotes the angle between two unit vectors, and is defined as the inverse cosine of their dot product.

Face-normal error metric: To quantify the performance of the proposed approach, we compute the L^2 face-normal error metric given by

$$E_f(\mathbb{M}, \widehat{\mathbb{M}}) = \frac{1}{\text{area}(\widehat{\mathbb{M}})} \sum_{\hat{\mathbf{t}}_j \in \widehat{\mathcal{T}}} \text{area}(\hat{\mathbf{t}}_j) \|\mathbf{n}(\mathbf{t}_j) - \mathbf{n}(\hat{\mathbf{t}}_j)\|, \tag{2.21}$$

where $\text{area}(\hat{\mathbf{t}}_j)$ is the area of $\hat{\mathbf{t}}_j$, and $\text{area}(\widehat{\mathbb{M}})$ is the total area of the denoised mesh.

Face quality metric: The quality of mesh faces can be measured using the ratio of the circumradius-to-minimum edge length given by

$$Q = \frac{1}{|\mathcal{T}|} \sum_{t \in \mathcal{T}} \frac{r_t}{\ell_t}, \tag{2.22}$$

where r_t and ℓ_t are the circumradius and minimum edge length of the associated triangle, respectively. In the ideal case, every face of the mesh should be an equilateral triangle with a quality index equal to $Q = 1/\sqrt{3}$.

The values of these metrics for our approach and the baselines methods are reported in Table 2.1. For fair comparison, we set the number of iterations to 5 for all the methods. Our approach yields better or competitive results in terms of E_o and E_f for all models. Moreover, the values of Q for our method are lower than those of the baseline methods. The L^2 face-normal errors for the left metacarpal, scaphoid, lunate, right metacarpal and left hamate are shown graphically in Figures 2.9-2.13. As can be seen in these figures, our method yields the best overall results, indicating the consistency with the subjective comparison.

Runtime Analysis

Most mesh denoising techniques perform filtering using a two-stage process by first filtering the face normals and then updating the vertex positions to match the filtered face normals, resulting in

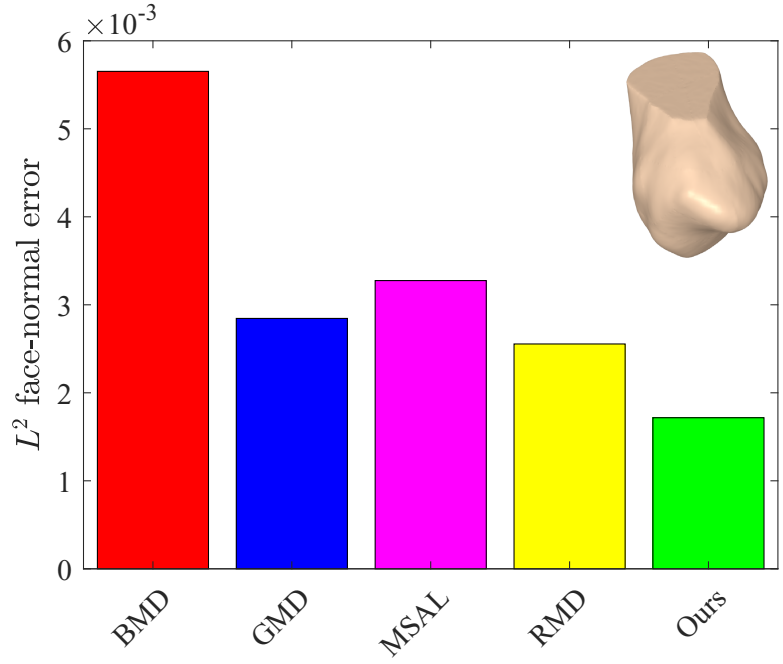

Figure 2.9: L^2 face-normal errors for the left metacarpal model.

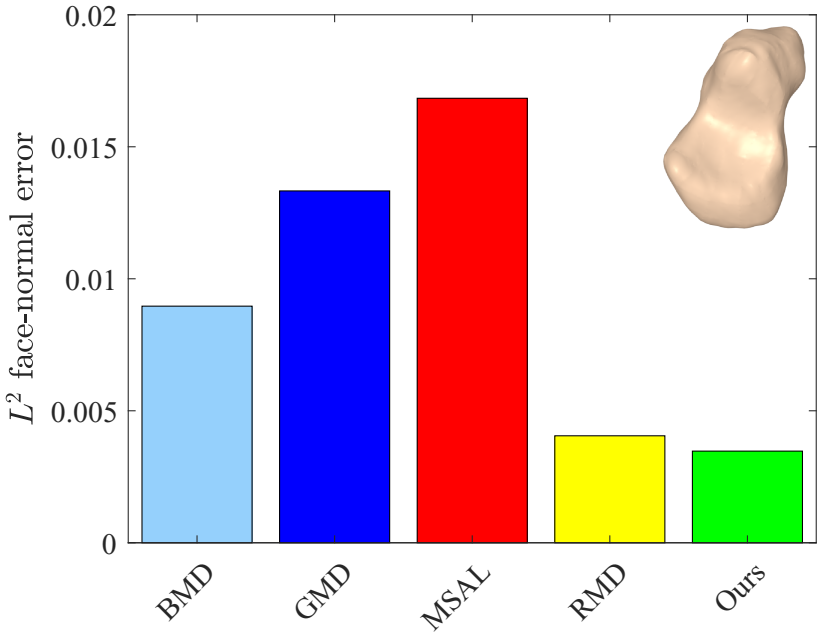

Figure 2.10: L^2 face-normal position errors for the scaphoid model.

a computationally expensive process, particularly for large 3D meshes. Our method is, however, fast and simple to implement. Table 2.2 shows the runtime of our algorithm for different carpal bone models. By comparison, the runtimes (in seconds) per iteration for RMD, which is the best performing baseline method, are 2.555, 2.3004, 2.292 and 2.167 for the right metacarpal, scaphoid,

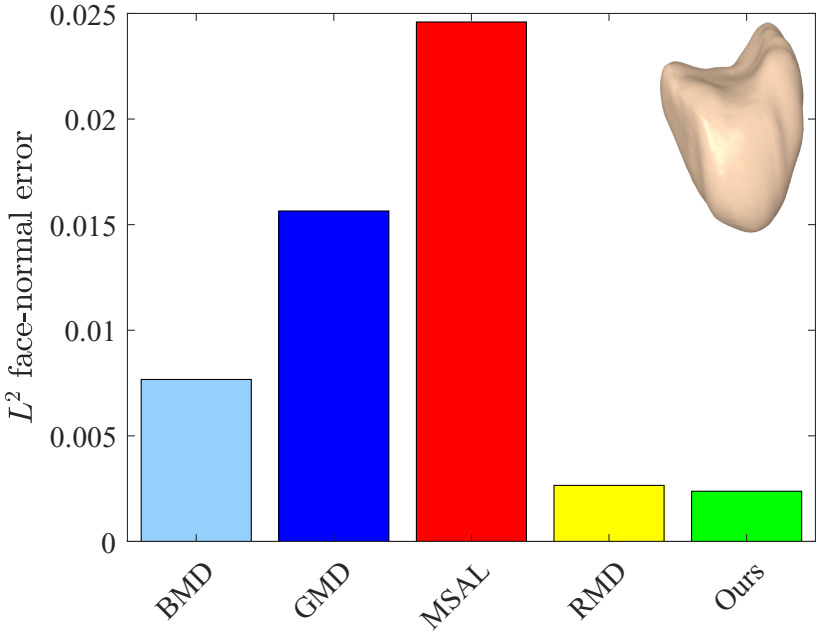

Figure 2.11: L^2 face-normal errors for the lunate model.

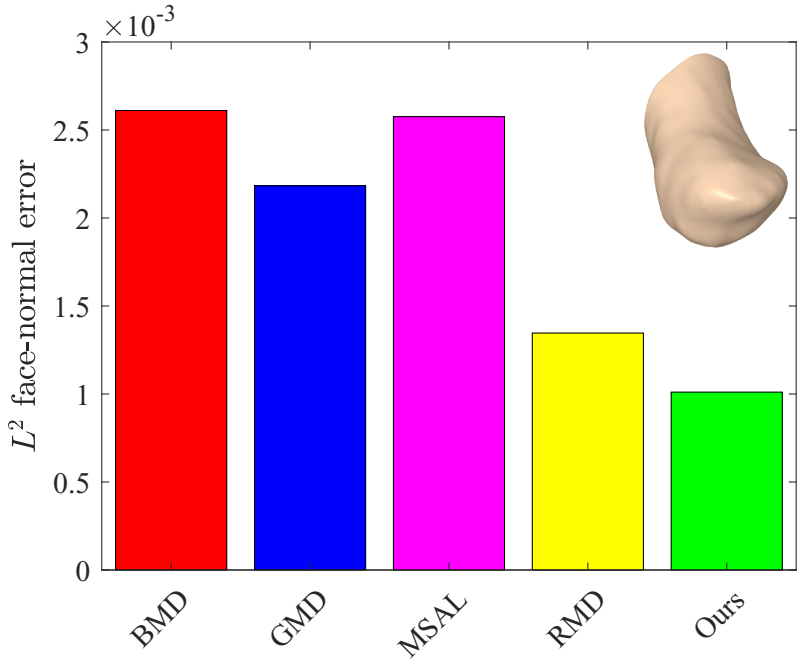

Figure 2.12: L^2 face-normal errors for the right metacarpal model.

left metacarpal and left hamate, respectively. This strongly indicates that our algorithm not only performs well at removing undesirable noise from bone surfaces, but is also computationally efficient.

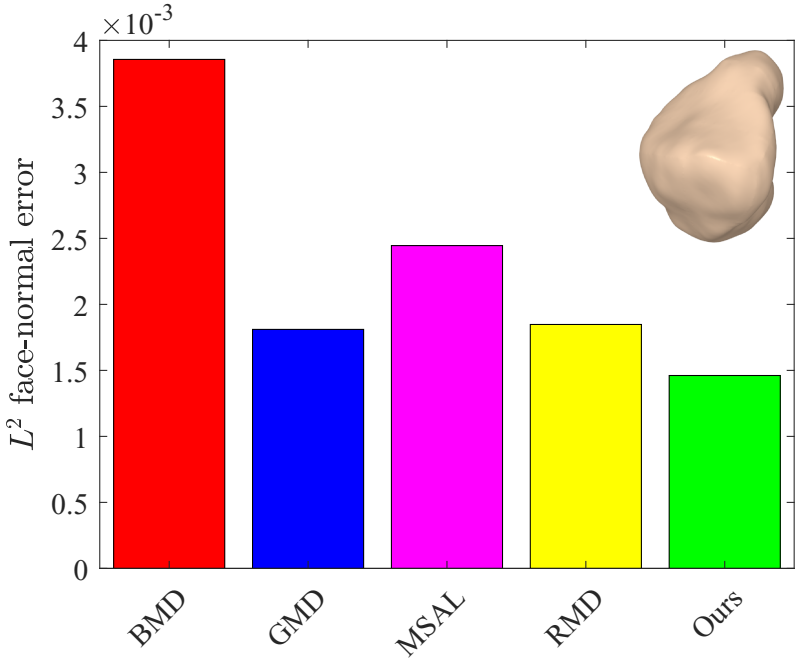

Figure 2.13: L^2 face-normal errors for the left hamate model.

Table 2.1: Quantitative comparison results.

Model	Method	E_o	$E_f \times 10^{-3}$	Q
	BMD	1.503	2.679	2.868
Right metacarpal	MSAL	1.506	2.495	6.852
$\|F\| = 27912$	GMD	1.503	2.183	7.973
$\|V\| = 13958$	RMD	1.503	2.686	4.700
	Ours	**1.470**	**1.011**	**1.623**
	BMD	1.506	7.770	5.226
Scaphoid	MSAL	1.530	16.838	2.247
$\|F\| = 29408$	GMD	1.457	13.332	7.940
$\|V\| = 14706$	RMD	**1.453**	3.976	1.888
	Ours	1.465	**2.966**	**1.678**
	BMD	1.510	5.652	2.449
Left metacarpal	MSAL	1.512	3.275	1.592
$\|F\| = 26858$	GMD	1.503	2.845	1.669
$\|V\| = 13431$	RMD	1.485	2.554	6.404
	Ours	**1.462**	**1.716**	**1.452**
	BMD	1.494	3.855	13.354
Left hamate	MSAL	1.506	2.445	5.597
$\|F\| = 28792$	GMD	**1.386**	1.811	4.663
$\|V\| = 14398$	RMD	1.418	1.847	2.826
	Ours	1.422	**1.461**	**1.702**

Table 2.2: Runtime (in seconds) per iteration and number of iterations used for denoising different models.

Model	Vertices	Faces	Time (sec.)	N_{iter}
Right metacarpal	13958	27912	0.284	5
Scaphoid	14706	29408	0.286	10
Left metacarpal	13431	26858	0.487	5
Left hamate	14398	28792	0.213	10

2.5 Conclusion

In this chapter, we presented a feature-preserving approach to surface denoising using a data-adaptive similarity in conjunction with matrix balancing. The proposed framework is formulated as a constrained minimization problem. The solution to this problem is estimated iteratively using the conjugate gradient method in an effort to recover sharp features from noisy surfaces. The qualitative and quantitative evaluation results demonstrate that our approach offers superior performance over existing mesh denoising techniques.

CHAPTER 3

Pediatric Bone Age Assessment

Bone age is an important measure for assessing the skeletal and biological maturity of children. Delayed or increased bone age is a serious concern for pediatricians, and needs to be accurately assessed in a bid to determine whether bone maturity is occurring at a rate consistent with chronological age. In this chapter, we introduce a unified deep learning framework for bone age assessment using instance segmentation and ridge regression. The proposed approach consists of two integrated stages. In the first stage, we employ an image annotation and segmentation model to annotate and segment the hand from the radiographic image, followed by background removal. In the second stage, we design a regression neural network architecture composed of a pre-trained convolutional neural network for learning salient features from the segmented pediatric hand radiographs and a ridge regression output layer for predicting the bone age. Experimental evaluation on a dataset of hand radiographs demonstrates the competitive performance of our approach in comparison with existing deep learning based methods for bone age assessment.

3.1 Introduction

Bone age assessment is a fundamental problem in the diagnosis and treatment of children and adolescents with suspected growth disorder [13]. It is often used in clinical practice by pediatricians for the children's skeletal maturation assessment in an effort to determine the difference between a child's bone age and a chronological age [62]. This difference influences the decisions taken by clinicians to predict the child's accurate age and often leads to errors in managing the diagnosis of children with skeletal dysplasias, and metabolic and endocrine disorders [63].

The assessment of bone age has been traditionally performed using the Greulich-Pyle (GP) and Tanner-Whitehouse (TW) methods [2, 3], which are based on left hand and wrist radiographs. In the GP method, the bones in the hand and wrist radiographs are compared to the bones of a standard atlas, while the TW method relies on a scoring system that examines the level of skeletal maturity for twenty selected regions of interest in specific bones of the hand-wrist and a numerical score is then assigned to selected hand-wrist bones depending on the appearance of certain well-defined maturity indicators. However, these manual methods suffer from substantial inter- and intra-observer variability, rely on trained radiologists, and are time consuming [64, 65]. To circumvent these issues, several automated bone age assessment approaches based on image processing and computer vision techniques have been proposed, including BoneXpert [14], which makes use of conventional radiographs of the hand according to the GP and TW methods. While BoneXpert is an automatic method, it does, however, require some manual analysis, particularly for X-ray images with low quality.

The recent trend in bone age assessment is geared toward automated methods using deep neural networks to learn features from hand radiographs at various levels of abstraction. This trend has been driven, in part, by a combination of affordable computing hardware, open source software, and the availability of large-scale datasets [66–68]. Several deep learning based models have been recently proposed to tackle the bone age assessment problem [15–17, 69–74], achieving good performance with substantially improved results. Spampinato *et al.* [69] presented a BoNet architecture comprised of five convolutional layers, one deformation layer, one fully connected later, and a linear scalar layer. They showed that BoNet outperforms fine-tuned convolutional neural networks for assessing bone age over races, age ranges and gender. Lee *et al.* [70] designed a deep learning model to detect and segment the hand and wrist prior to performing bone age assessment with a fine-tuned convolutional neural network. Larson *et al.* [71] applied a fifty-layer residual network to estimate bone age, achieving comparable performance to that of trained human reviewers. However, their model is not effective at predicting the bone age of patients younger than two years.

In this chapter, we propose an integrated deep learning based framework, called RidgeNet, for bone age assessment using instance segmentation and ridge regression. We develop a regression network architecture for bone age assessment using a pre-trained deep learning model in conjunction with a regularized regression output layer. The main contributions of this work can be summarized as follows:

- We present an image annotation and segmentation model to annotate and segment the hand from the radiographic image with a minimum number of annotated images, followed by background removal.

- We leverage the power of transfer learning with fine-tuning to learn salient features from the segmented radiographs.
- We design a regression neural network architecture with a ridge regression output layer for predicting the bone age.
- We show through extensive experiments the competitive performance of the proposed approach in comparison with baseline methods.

The rest of this chapter is organized as follows. In Section 2, we review important relevant work. In Section 3, we introduce a two-stage approach for bone age assessment using instance segmentation and ridge regression. In the first stage, we employ an image annotation and instance segmentation model to extract and separate different regions of interests in an image. In the second stage, we leverage the power of transfer learning by designing a deep neural network with a ridge regression output layer. Section 4 presents experimental results to demonstrate the competitive performance of our approach compared to baseline methods. Finally, Section 5 concludes the chapter and points out future work directions.

3.2 Related Work

The basic goal of bone age assessment is to evaluate the biological maturity of children. To achieve this goal, various bone age prediction methods based on deep learning have been proposed. Tong *et al.* [72] presented an automated skeletal bone age assessment model using convolutional neural networks and support vector regression with multiple kernel learning by combining heterogeneous features from X-ray images, race and gender. Van Steenkiste *et al.* [15] integrated a pre-trained convolutional neural network with Gaussian process regression in order to refine the estimated bone age by exploiting variations in the predictions. The Gaussian process regression is used to estimate a vector of prediction scores for rotated and mirror images of a single radiograph. Iglovikov *et al.* [16] proposed deep learning based regression and classification models using convolutional neural networks by applying image segmentation via U-Net, a fully convolutional network that extracts regions of interest (ROIs) from images and predicts each pixel's class [38]. With the exception of the last two layers, the regression model proposed in [38] is similar to the classification one, where each bone age is assigned a class. In the last layer of the classification model, probabilities obtained from the softmax layer are multiplied by a vector of bone ages uniformly distributed over integer values. Wu *et al.* [17] designed a network architecture consisting of an instance segmentation model and a residual attention network. The instance segmentation model segments the

hands from X-ray images to avoid the distractions of other objects, while the residual attention network forces the network to focus on the main components of the X-ray images. Liu *et al.* [74] replaced the encoder of U-Net with a pre-trained convolutional neural network to perform image segmentation, followed by applying a ranking learning technique instead of regression to assess bone age. Similarly, Pan *et al.* [75] introduced a U-Net based model, which consists of image segmentation, feature extraction, and ensemble modules. More recently, Liu *et al.* [76] proposed a bone age assessment model, which is trained on multiclassifiers based on ensemble learning, to predict the optimal segmentation threshold for hand mask segmentation. In [77], a region-based feature connected layer from the essential segmented region of a hand X-ray is introduced in order to predict bone age using deep learning models.

While these deep learning based approaches have yielded competitive results in bone age assessment, they suffer, however, from high model complexity, often require a pre-processing image alignment step, and do not distinguish between important from less-important predictors in the regression model. In addition, the U-Net architecture and its variants suffer from the large semantic gap between the low- and high-level features of the encoder and decoder subnetworks, leading to fusing semantically dissimilar features and hence resulting in blurred feature maps throughout the learning process and also adversely affecting the output segmentation map by under- and/or over-segmenting regions of interest (ROIs).

3.3 Method

Bone age prediction is typically achieved by extracting ROI features from pediatric hand images, followed by using a learning model to estimate the bone age of these radiographs. Our algorithm is divided into two stages. In the first stage, we carried out an image preprocessing step for radiographs using image annotation and segmentation. In the second stage, we trained the proposed deep learning model using the segmented radiographs to evaluate the proposed model's performance.

3.3.1 Image Annotation and Segmentation Model

This preprocessing step aims at extracting regions of interest from radiographs using image annotation and instance segmentation.

Image Annotation: We use the VGG image annotator[1] to define regions in an image and create textual descriptions of those regions for image segmentation purposes. The annotations can be

[1] http://www.robots.ox.ac.uk/ vgg/software/via/via-2.0.4.html

manually done using rectangles, circles, ellipses, polygons, lines, or points, and then converted to common objects in context (COCO) dataset format, which is the commonly used format for object detection and instance segmentation algorithms. As shown in Figure 3.1, the annotated hand is obtained using a polygonal region shape.

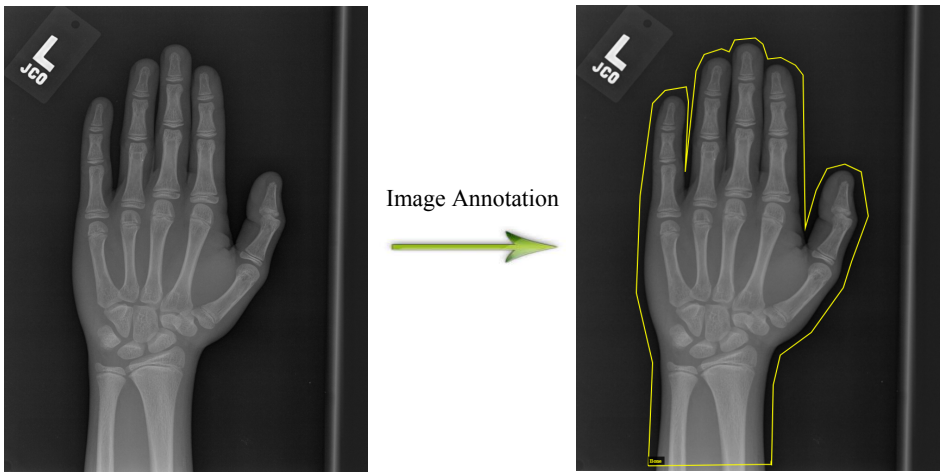

Figure 3.1: Image annotation using VGG image annotator.

Instance Segmentation: Instance segmentation refers to the task of detecting and delineating each distinct object of interest appearing in an image, and includes identification of boundaries of the objects at the detailed pixel level. Several instance segmentation techniques have been recently proposed, including mask region-based convolutional neural network (Mask R-CNN), which locates each pixel of every object in the image instead of the bounding boxes [40]. The input to the Mask R-CNN algorithm is an image and the output is a bounding box and a mask that segment each object in the image, as shown in Figure 3.2. Mask R-CNN consists of two main stages. The first stage scans the image and generates proposals about the regions where there might be an object. The second stage classifies the proposals and generates bounding boxes and masks. These two stages are connected to a backbone network that uses residual neural network in conjunction with feature pyramid network for feature extraction. The feature pyramid network constructs a high-level feature pyramid and recognizes objects at different scales. In addition to using a region-of-interest (RoI) pooling, which performs max pooling on inputs of non-uniform sizes and produces a small feature map of fixed size, Mask R-CNN employs an ROIAlign layer that aligns the extracted features with the input in a bid to avoid any quantization of the RoI boundaries or bins.

For radiograph segmentation, we leverage Mask R-CNN that was trained on the COCO dataset. The given annotated images are split into two disjoint subsets: the training set for learning, and

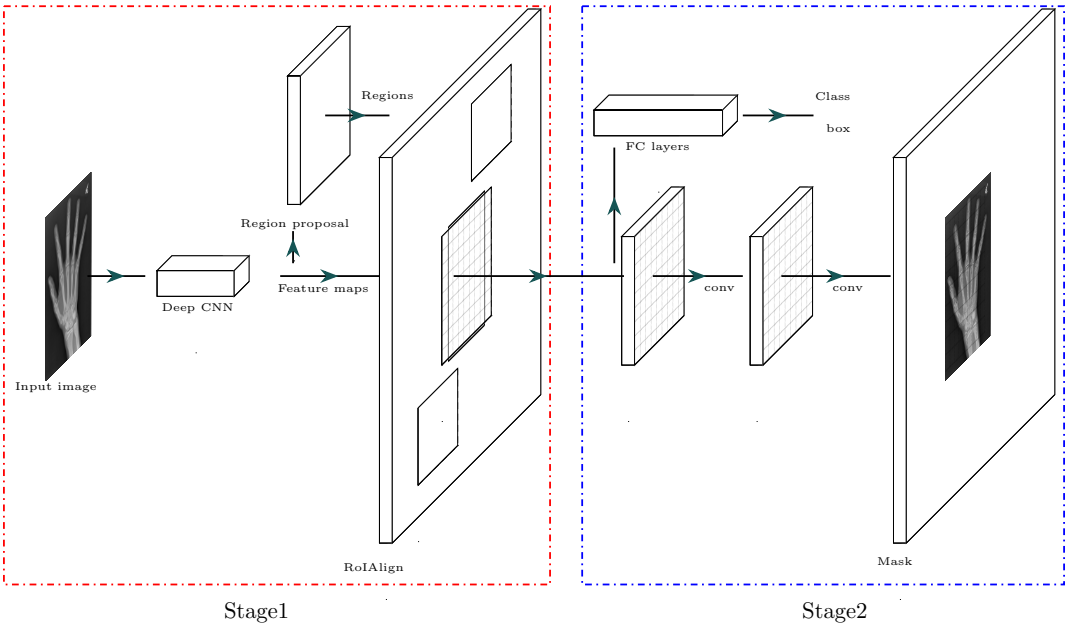

Figure 3.2: Instance segmentation using Mask R-CNN.

the test set for testing. We used the pre-trained weights of some layers of Mask R-CNN as initial weights. More specifically, we trained the region proposal network, classifier and mask heads layers of Mask R-CNN and freezed the other hidden layers to speed-up the training of the network. The trainable weights generated after training are used to perform instance segmentation. The flowchart for image segmentation and background removal is depicted in Figure 3.3. As can be seen, the detected part of the image (whole hand) is classified as a foreground with a very high accuracy, while the rest of the radiograph image is classified as a background. Note that the red colored hand and the dotted rectangle represent the bounding box and mask, respectively.

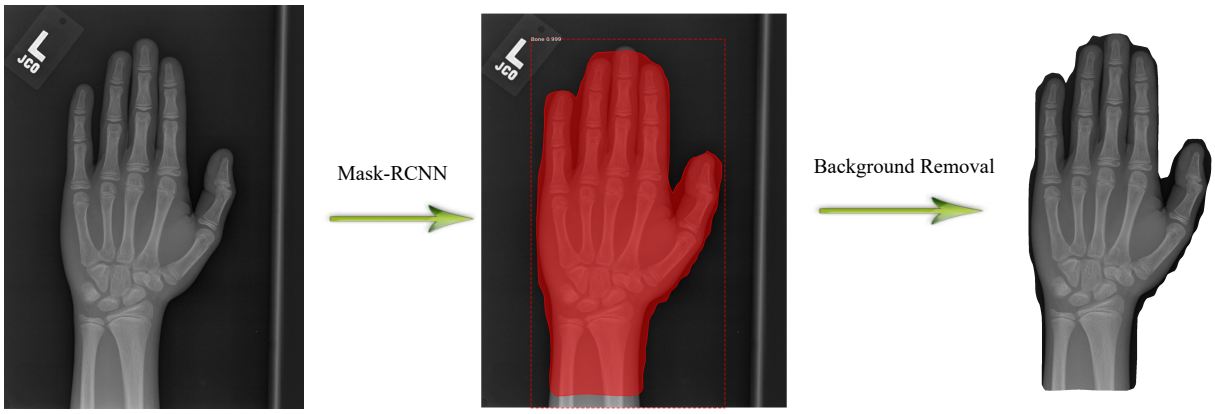

Figure 3.3: Image segmentation using Mask R-CNN, followed by background removal.

3.3.2 Bone Age Assessment Model

The proposed bone age assessment model uses the pre-trained VGG-19 convolutional neural network with a ridge regression output layer, as illustrated in Figure 3.4. The VGG-19 network consists of 19 layers with learnable weights: 16 convolutional layers, and 3 fully connected layers [33]. Each convolutional layer uses a 3×3 kernel with stride 2 and padding 1. A 2×2 max-pooling is performed with stride 2 and zero padding. As shown in Figure 3.4, the proposed architecture consists of five blocks of convolutional layers, followed by a global average pooling (GAP), a fully connected layer of 1024 neurons, a dropout layer, a ridge regression layer, and single regression output. Each of the first and second blocks is comprised of two convolutional layers with 64 and 128 filters, respectively. Similarly, each of the third, fourth and fifth blocks consists of four convolutional layers with 256, 512, and 512 filters, respectively. The GAP layer, which is widely used in object localization, computes the average output of each feature map in the previous layer and helps minimize overfitting by reducing the total number of parameters in the model. GAP turns a feature map into a single number by taking the average of the numbers in that feature map, and helps identify where deep neural networks pay attention. Similar to max pooling layers, GAP layers have no trainable parameters and are used to reduce the spatial dimensions of a three-dimensional tensor.

Ridge regression, on the other hand, is a regularized regression method that shrinks the estimated coefficients towards zero. More specifically, given a response vector $\mathbf{y} \in \mathbb{R}^n$ and a predictor matrix $\mathbf{X} \in \mathbb{R}^{n \times p}$, the ridge regression coefficients are defined as

$$\hat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta} \in \mathbb{R}^p} \left\{ \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 + \lambda \|\boldsymbol{\beta}\|_2^2 \right\}, \quad \lambda \geq 0 \tag{3.1}$$

where λ is a tuning parameter that controls the strength of the penalty term and decides the shrinkage amount of the coefficients. A larger value of λ indicates more shrinkage. In practice, the hyperparameter λ is often estimated using cross-validation.

The solution to the ridge regression equation (3.1) is given by

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^\intercal \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^\intercal \mathbf{y}, \tag{3.2}$$

which reduces to the linear regression estimate when $\lambda = 0$. Unlike linear regression which does not differentiate "important" from "less-important" predictors in a model, ridge regression reduces model complexity and prevents over-fitting by producing new estimators that are shrunk closer to the "true" parameters.

The ridge regression layer computes the mean square error (MSE) given by

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2, \tag{3.3}$$

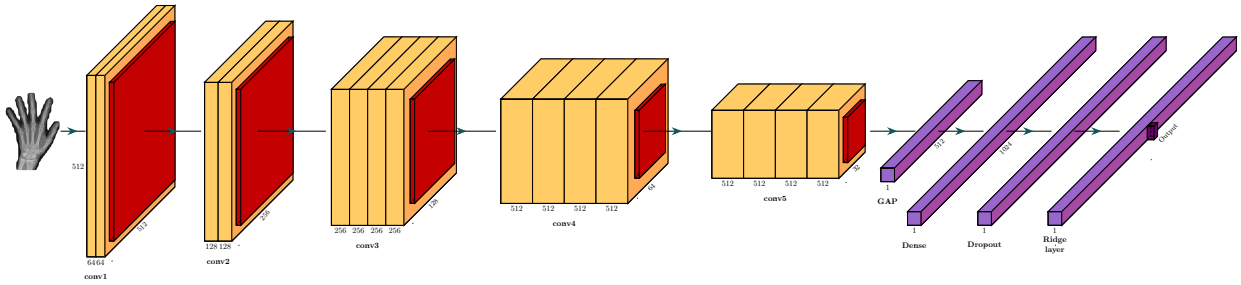

Figure 3.4: Architecture of the proposed regression network.

where n is the number of responses, y_i is the target output, and $\hat{y}_i$ is the network's prediction for response i.

The goal of the ridge regression layer is to find the optimal values of the regression coefficients, while avoiding the multicollinearity issue that frequently occurs in multiple linear regression. Multicollinearity in a regression model is a statistical phenomenon that arises when some predictor variables in the model are correlated with other predictor variables, leading to increased variance of the regression coefficients and hence making them unstable. In order to mitigate the multicollinearity issue, ridge regression is usually used not only to reduce model complexity, but also to prevent over-fitting by adding a regularization term in an effort to ensure a smaller variance in resulting parameter estimates. Motivated by these nice properties of ridge regression, we design a ridge regression layer and incorporate it into our proposed network architecture with the goal of circumventing both the multicollinearity and over-fitting issues.

3.4 Experiments

In this section, we conduct extensive experiments to evaluate the performance of the proposed RidgeNet model in bone age assessment. The effectiveness of RidgeNet is validated by performing a comprehensive comparison with several baseline methods.

Dataset: The effectiveness of RidgeNet is evaluated on the Radiological Society of North America (RSNA) bone age dataset [68]. RSNA consists of 14,236 hand radiographs of both male and female patients. The radiographs were acquired from Stanford Children's and Colorado Children's Hospitals at different times and under different conditions, and there are 12,611 images in the training set, 1,425 images in the validation set, and 200 images in the test set. Sample hand radiographs from the RSNA dataset are shown in Figure 3.5. As can be seen, the images were acquired at different times and under different conditions with varying size, background, contrast, brightness, and hand orientation. The training set contains 5778 radiographs of female patients and 6833 ra-

diographs of male patients. As shown in Figure 3.6, the bone ages for male and female patients are not uniformly distributed.

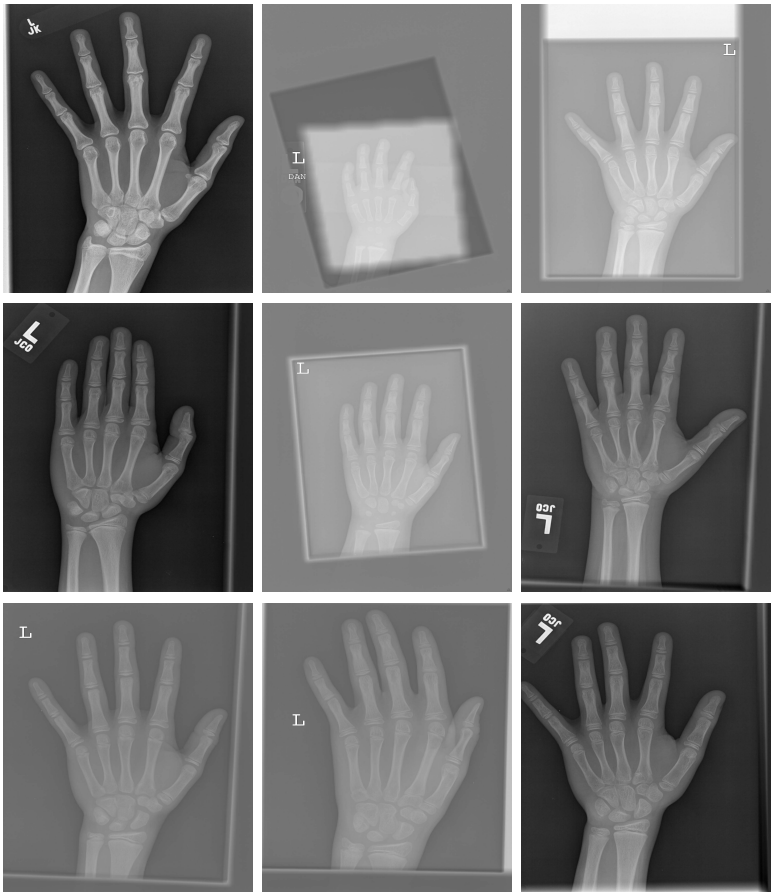

Figure 3.5: Sample radiograph images from the RSNA dataset.

Implementation details: All the experiments were performed on Google Colaboratory, a free Jupyter notebook environment with 2 CPUs Intel(R) Xeon(R) 2.2GHz, 13GB RAM, and an NVIDIA Tesla K80 GPU accelerator. The algorithms were implemented in Python using Keras with Tensorflow backend. We used data augmentation to improve network accuracy and avoid overfitting by expanding the size of a training dataset. This is usually done by creating modified versions of the input images in the dataset through random transformations, including horizontal and vertical flip, brightness and zoom augmentation, horizontal and vertical shift augmentation, and rotation. For image annotation, we chose 80 images with different scales and orientations, and we used a polygon region shape to annotate the hand in each radiograph. Since we used the COCO dataset format for annotation, the annotated images are saved in JavaScript Object Notation (JSON) format for our custom dataset of segmented hand radiographs.

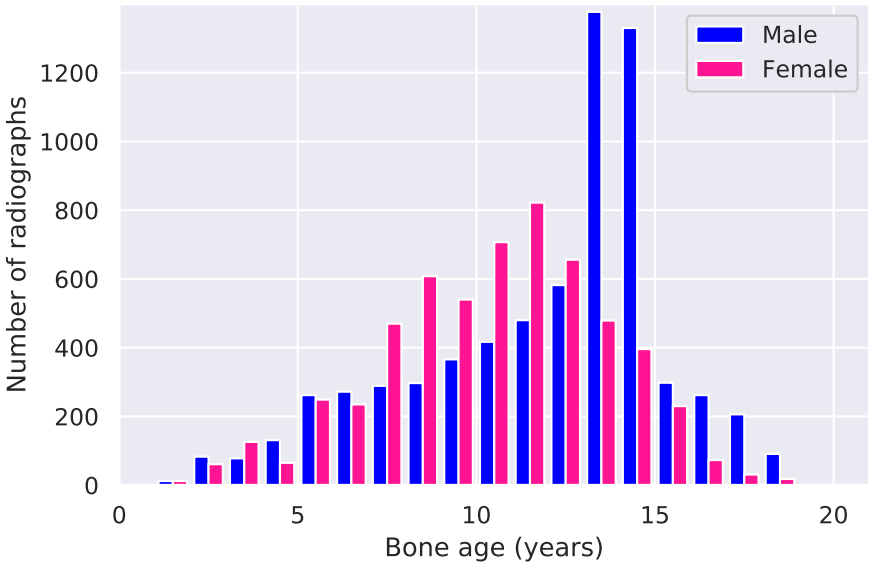

Figure 3.6: Bone age distributions of male and female patients in the training set.

For feature extraction, we use the pre-trained VGG-19 convolutional neural network with input image size $512 \times 512 \times 3$, and rectified linear unit (ReLU) as an activation function. For ridge regression, the regularization parameter is $\lambda = 10^{-4}$, which was obtained via cross-validation. The proposed regression network was trained for 160 epochs using Adam optimizer with initial learning rate of 10^{-4} and batch size of 32, and using the mean square error as a loss function. The learning rate was multiplied by factor of 0.8 automatically when validation loss plateaued for 3 epochs. The value of MSE is computed as the average of the squared differences between the predicted and actual values.

The MSE values are recorded at the end of each epoch on the training and/or validation sets. The performance of the RidgeNet model over training epochs on the training and validation sets is evaluated using the mean absolute error (MAE) metric, as shown in Figure 3.7. As can be seen, the RideNet model has comparable performance on both training and validation sets, indicating the higher predictive accuracy of the proposed model.

Performance evaluation metrics: The performance of a regression model is usually assessed by applying it to test data with known target values and comparing the predicted values with the known values. To this end, we use mean absolute error (MAE), root mean square error (RMSE), and root mean squared percentage error (RMSPE) as evaluation metrics, which are defined as follows:

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i| \tag{3.4}$$

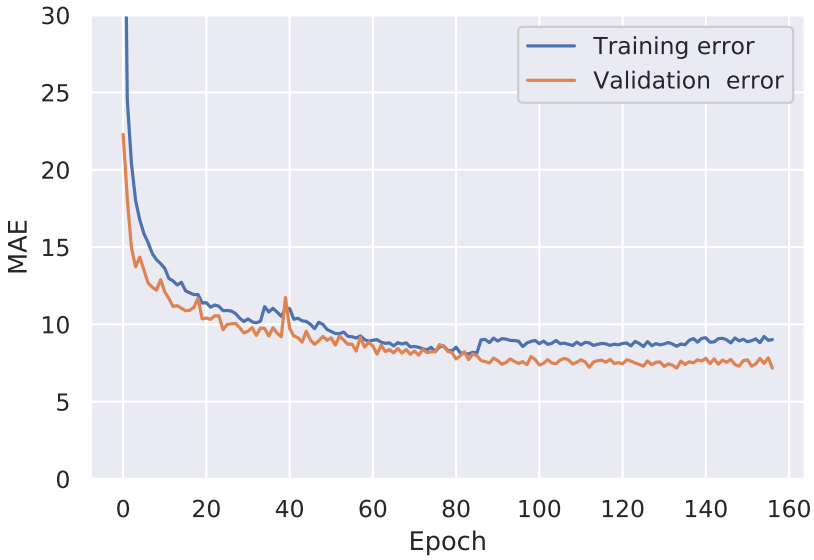

Figure 3.7: Model training history.

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2} \tag{3.5}$$

$$\text{RMSPE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\frac{y_i - \hat{y}_i}{y_i}\right)^2} \tag{3.6}$$

where N is the number of samples in the test set, y_i is the actual (ground truth) value, and $\hat{y}_i$ is the model's predicted value. A small value of an evaluation metrics indicates a better performance of the model.

Baseline methods: We compared the effectiveness of the proposed model with several deep learning based approaches, including a pre-trained convolutional neural network with Gaussian process regression [15], an instance segmentation model with residual attention network [17], regression/classification models [16], and U-Net based model [75]. The regression model presented in [16] is similar to the classification model, expect for the lat two layers and also the fact that a bone age is assigned a class. The penultimate layer is a softmax layer that returns a vector of probabilities for all the classes. In the last layer, these probabilities are multiplied by a vector of bone ages uniformly distributed over an interval of length equal to the total number of classes. The U-Net based model [75] consists of image segmentation, feature extraction, and ensemble modules.

3.4.1 Results

In this subsection, we report the prediction results obtained by our proposed approach on the RSNA bone age test set, and provide a comparison analysis with baseline methods. Figures 3.8-3.10 show the predicted bone age versus the actual bone age for both genders, males, and females, respectively, on the test set. The solid line depicts the actual bone age, while the dot points represent the predicted values. As can be seen, the predicted values align pretty well along the solid line, indicating a strong agreement between the actual bone age and predicted one. It is important to point out that the plot for both genders in Figure 3.8 shows a noticeable variation in skeletal age prediction between 84 and 156 months. This variation is due largely to the fact that girls mature faster than boys during the early and mid-puberty stage (7- to 13-year old girls compared with 9- to 14-year old boys), and hence skeletal maturation changes occur earlier in females than in males. Notice also that there is little variation in skeletal age prediction during infancy (birth to 10-month old girls and birth to 14-month old boys), toddlers (10-month to 2-year old girls and 10-month to 3-year old boys), pre-puberty (2- to 7-year old girls and 3- to 9-year old boys) compared to late puberty (13- to 15-year old girls and 14- to 16-year old boys) and post-puberty (14- to 17-year old girls and 17- to 19-year old boys).

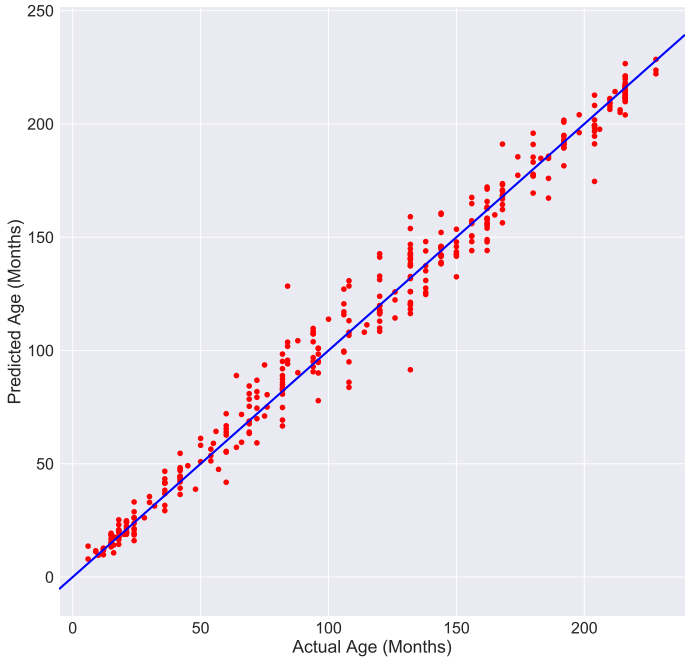

Figure 3.8: Predicted bone age vs. actual bone age for both genders on the RSNA test set.

The bar plots of the MAE values obtained by RidgeNet and the baseline methods for both genders are displayed in Figure 3.11, which shows that the proposed approach performs the best,

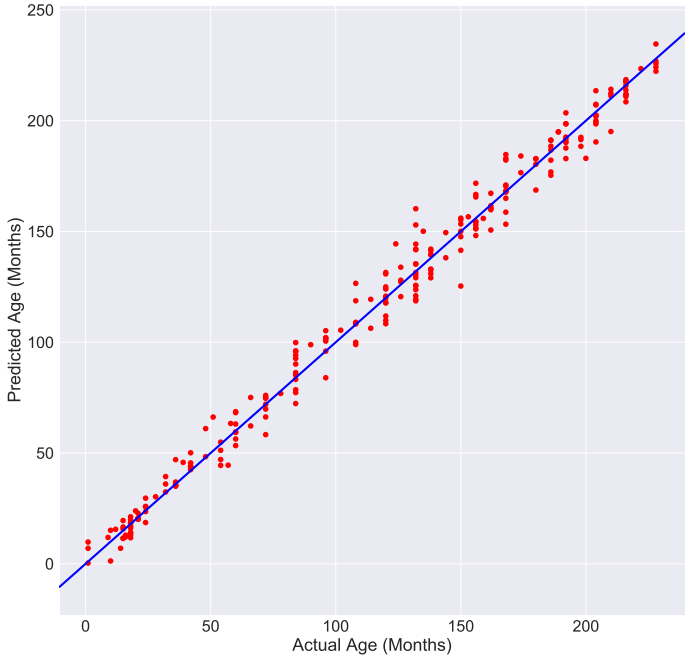

Figure 3.9: Predicted bone age vs. actual bone age for male patients on the RSNA test set.

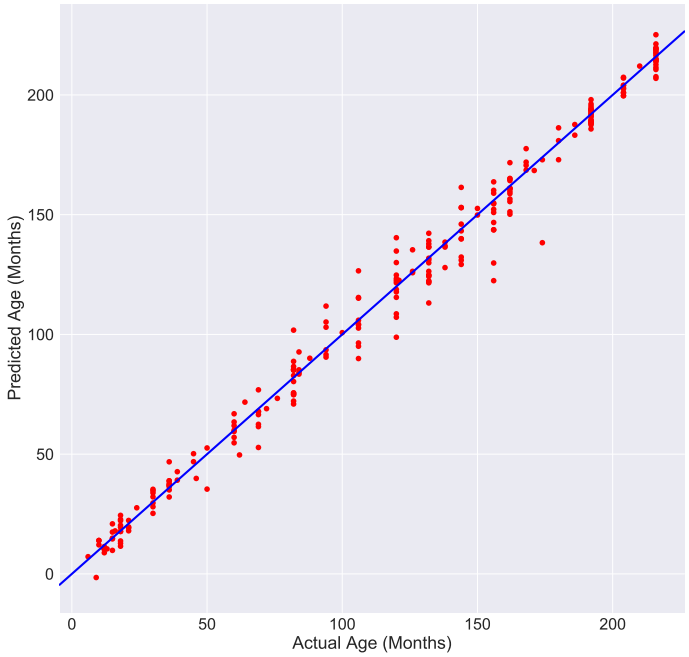

Figure 3.10: Predicted bone age vs. actual bone age for female patients on the RSNA test set.

yielding the lowest MAE of 6.38 months. Similarly, the bar plots shown in Figures 3.12 and 3.13 for male and female patients, respectively, indicate that RidgeNet substantially outperforms both the regression and classification models. As can be seen in these figures, the proposed approach yields the best overall results. In addition, it is worth pointing out that gender-specific regression

by RidgeNet yields lower MAE values, with a MAE of 5.27 months for females and a MAE of 3.75 months for males compared to a MAE of 6.38 months for both sexes.

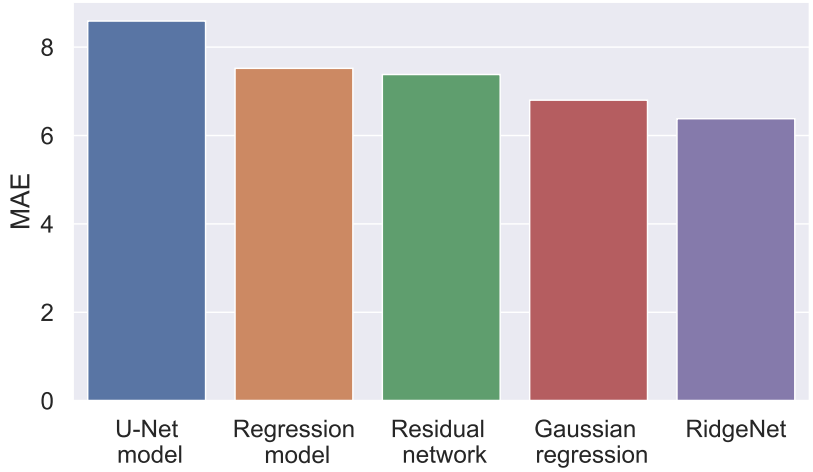

Figure 3.11: MAE results for RidgeNet and baseline methods for both genders on the RSNA test set.

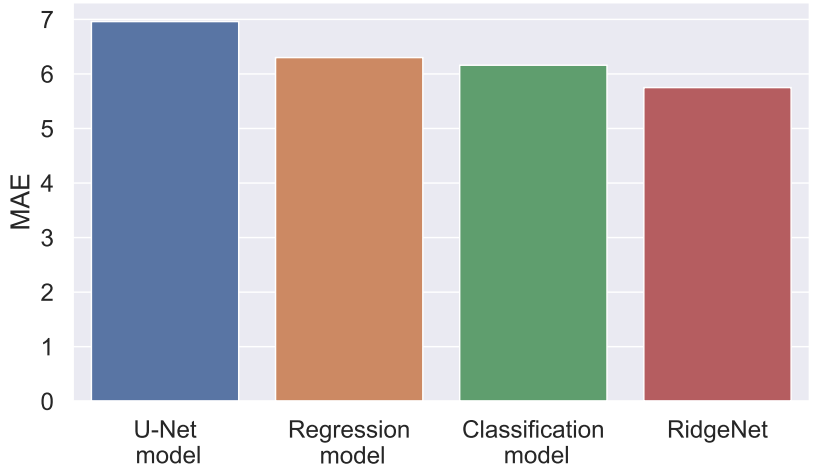

Figure 3.12: MAE results for RidgeNet and baseline methods for male patients on the RSNA test set.

The evaluation results using the RMSE and RMSPE metrics are shown in Table 3.1. As can be seen, RidgNet yields the lowest RMSPE value in the case of both genders.

Figure 3.14 shows the actual and predicted bone ages of some segmented images from the test set. As can be seen, the predicted bone ages are quite comparable to the actual bone ages, indicating the effectiveness of the proposed RidgeNet model in bone age assessment.

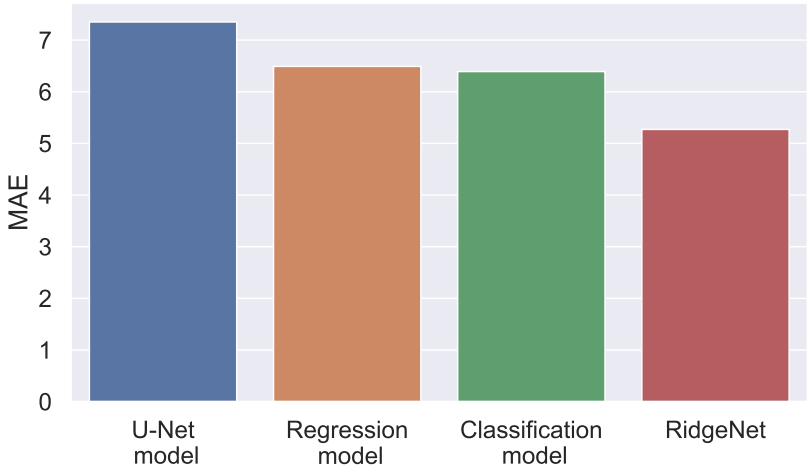

Figure 3.13: MAE results for RidgeNet and baseline methods for female patients on the RSNA test set.

Table 3.1: Evaluation results for RidgeNet on the RSNA test set using the RMSE and RMSPE metrics.

Gender	RMSE	RMSPE
Both	8.70	2.71
Males	7.00	3.06
Females	7.49	3.08

3.4.2 Feature Visualization and Analysis

Understanding and interpreting the predictions made by a deep learning model provides valuable insights into the input data and the features learned by the model so that the results can be easily understood by human experts. To visually explain the predictions obtained by the proposed RidgeNet model, we apply Smooth Grad-CAM++, an enhanced gradient weighted class activation mapping [78] that highlights the most influential features affecting the predictions of the model on various radiographs from different skeletal maturation stages, including pre-puberty, early and mid-puberty, late puberty, and post-puberty. Gradient weighted class activation mapping produces heat maps via a linear weighted combination of the activation maps to highlight discriminative image regions, indicating where the deep neural network bases its predictions. The Smooth Grad-CAM++ method combines concepts from Grad-CAM [79], SmoothGrad [80] and Grad-CAM++ [81] to produce improved, visually sharp maps for specific layers, subset of feature maps and neurons of interest.

Figure 3.15 shows the class activation maps for RidgeNet using Smooth Grad-CAM++ for female (top) and male (bottom) patients at the pre-puberty, early and mid-puberty, late puberty, and

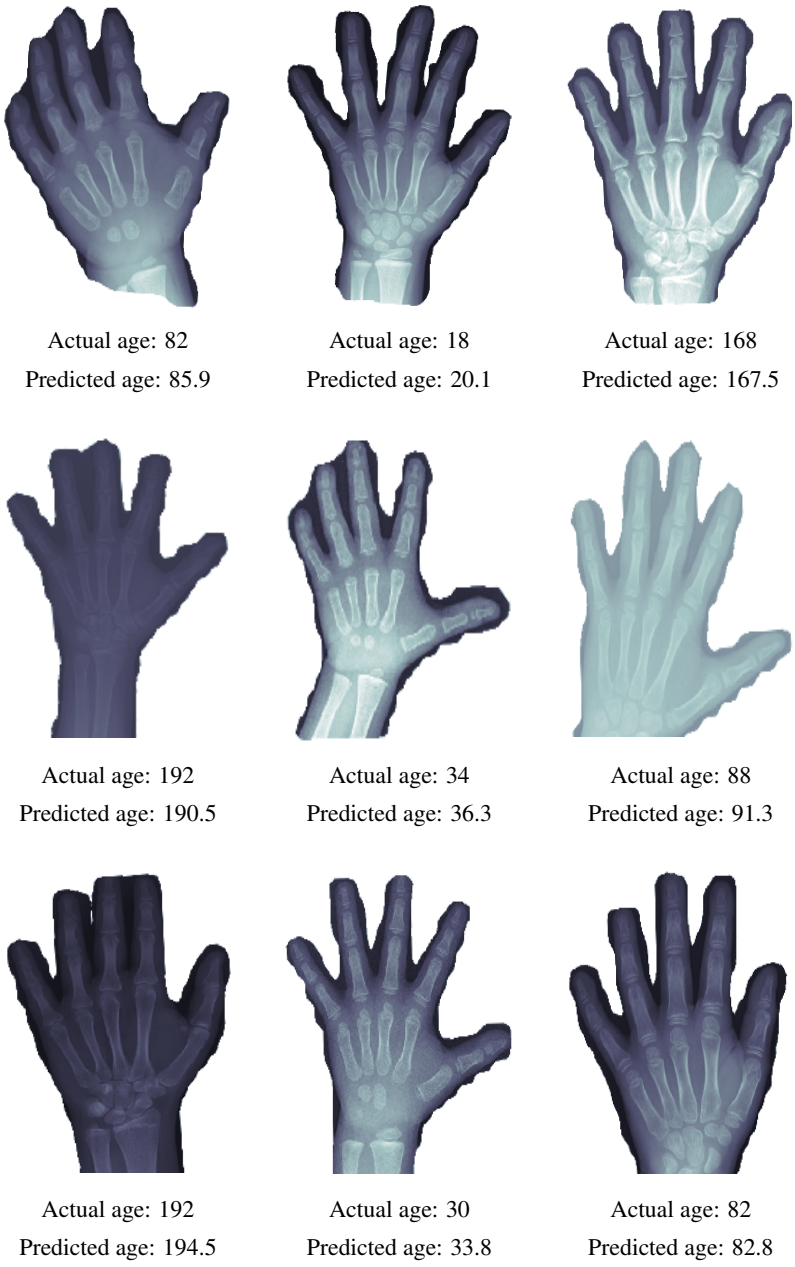

Figure 3.14: Actual and predicted bone age of sample images from the test RSNA dataset. The first row displays images of both genders, while the second and the third row show the images of males and females, respectively

post-puberty stages. As can be seen, Smooth Grad-CAM++ highlights the regions of the radiographs that play a significant role in bone age assessment for different age categories. The red regions contribute the most to the predictions obtained by RidgeNet, and the deep blue color indicates the lowest contribution. In the early and mid-puberty stage, for instance, the class activation map emphasizes the metacarpal bones and proximal phalanges. In the post-puberty stage, the Rid-

geNet model focuses on carpal bones, meaning that they have the greatest impact on the regression results. These results are consistent with what radiologists consider the most suitable indicators of skeletal maturity during the different phases of postnatal development [13].

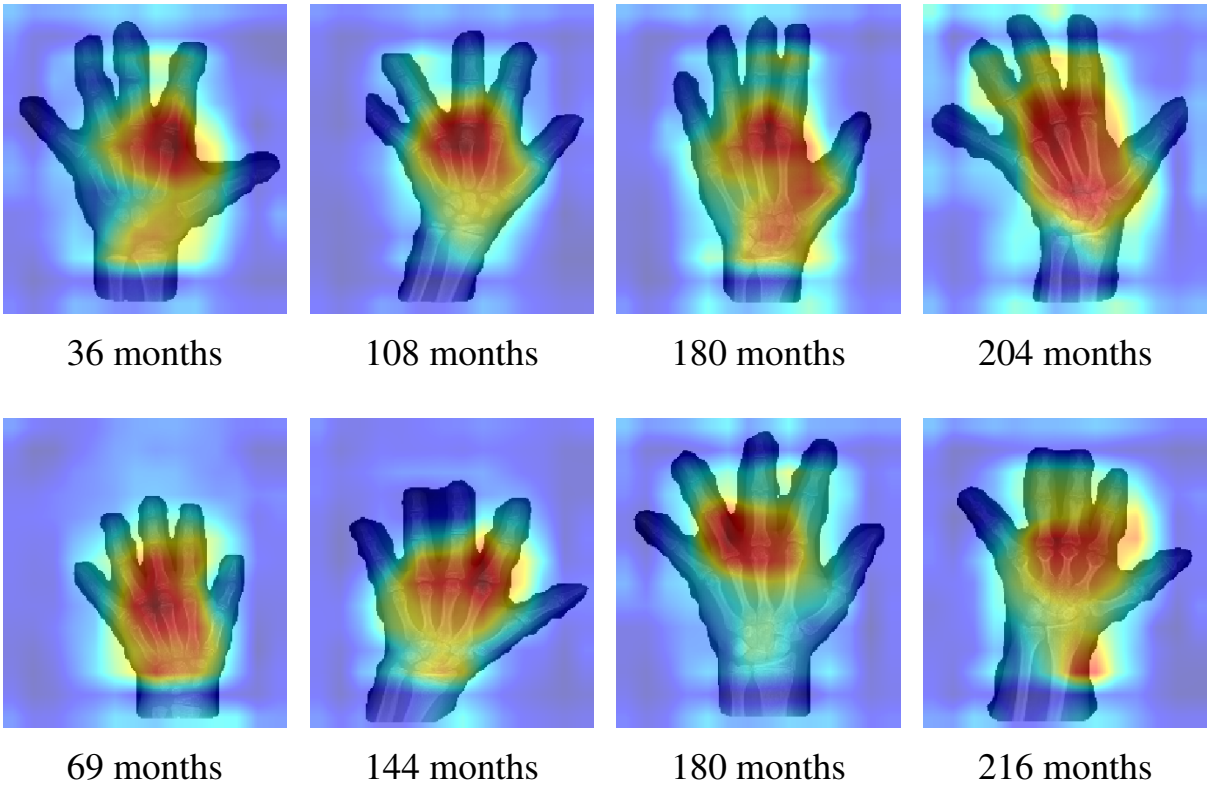

Figure 3.15: Smooth Grad-CAM++ heat maps for female (top) and male (bottom) patients at the pre-puberty, early and mid-puberty, late puberty, and post-puberty stages.

3.4.3 Age Assessment on chest X-rays

In order to further assess the effectiveness of our age prediction approach, we evaluate the proposed model on the National Institutes of Health (NIH) chest X-ray dataset [82], which is comprised of 112,120 X-ray images with different ages from 30,805 unique patients. We evaluate the proposed model on 7,240 patients in the age range of 0 to 20 years old. The dataset is randomly partitioned into training (70%), validation (10%) and testing (20%). Sample X-ray images from the NIH datsets are shown in Figure 3.16, and the evaluation results using the MAE, RMSE and RMSPE metrics are reported in Table 3.2.

Figure 3.17 shows the actual and predicted ages of some images from the NIH chest test set. As can be seen, the proposed model yields relatively good results, which we plan to further improve as future work.

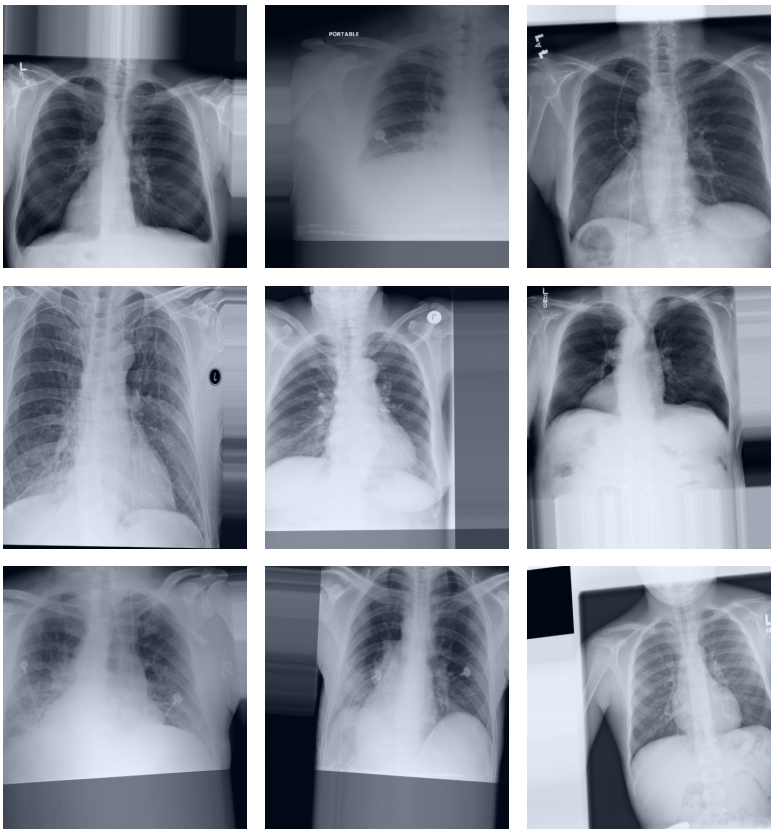

Figure 3.16: Sample X-ray images from the NIH chest dataset.

Table 3.2: Evaluation results for RidgeNet on the NIH chest X-ray test set using the MAE, RMSE and RMSPE metrics.

Gender	MAE	RMSE	RMSPE
Both	52.21	61.03	2.56
Males	46.61	55.96	2.75
Females	44.20	53.79	2.13

3.5 Conclusion

In this chapter, we presented a two-stage approach for bone age assessment using instance segmentation and ridge regression. The proposed framework uses instance segmentation to extract a region of interest from radiographs and background removal to avoid all extrinsic objects, followed by a regression network architecture with a ridge regression output layer that returns a single, continuous value. We also used dropout regularization to improve the generalization ability, and we shuffled the training data before each epoch to help detect overfitting issues and increase the model performance. We showed through extensive experiments on the RSNA dataset that the proposed RidgeNet model significantly outperforms existing deep learning based methods for bone age as-

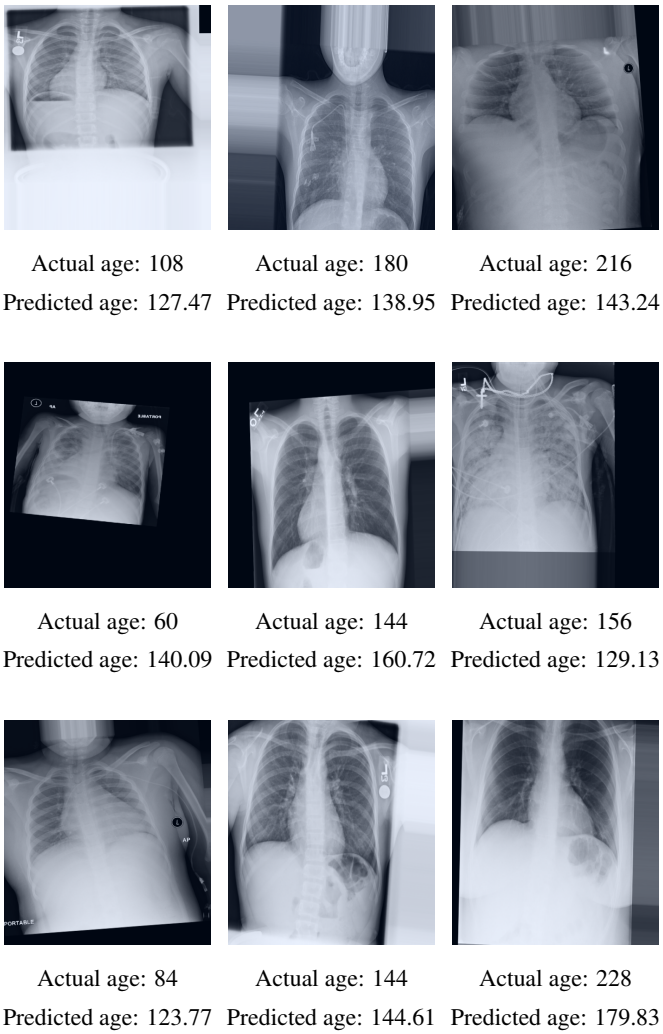

Figure 3.17: Actual and predicted age of sample image from the test NIH dataset in months. The X-ray images of both genders are shown in the first row, while the second and third rows display the X-ray images for males and females, respectively

sessment, achieving the lowest mean absolute error for male and female patients, as well as for both genders.

CHAPTER 4

Classification of Developmental and Brain Disorders

While graph convolution based methods have become the de-facto standard for graph representation learning, their applications to disease prediction tasks remain quite limited, particularly in the classification of neurodevelopmental and neurodegenerative brain disorders. In this chapter, we introduce an aggregator normalization graph convolutional network by leveraging aggregation in graph sampling, as well as skip connections and identity mapping. The proposed model learns discriminative graph node representations by incorporating both imaging and non-imaging features into the graph nodes and edges, respectively, with the aim to augment predictive capabilities. We benchmark our model against several recent baseline methods on two large datasets, ABIDE and ADNI, for the prediction of autism spectrum disorder and Alzheimer's disease, respectively. Experimental results demonstrate the competitive performance of our approach in comparison with recent baselines in terms of several evaluation metrics, achieving relative improvements of 50% and 13.56% in classification accuracy over graph convolutional networks on ABIDE and ADNI, respectively.

4.1 Introduction

Understanding how the brain develops is vital to designing prediction models and formulating treatments for a variety of developmental disorders and degenerative neurological diseases such as autism spectrum disorder and Alzheimer's disease, which are devastating illnesses that have touched the lives of millions of families around the world, not only in personal anguish, but also in soaring healthcare costs [83]. Autism spectrum disorder is a neurodevelopmental disability that

affects how a person communicates, learns and socializes with others, whereas Alzheimer's disease is a chronic neurodegenerative brain disorder that slowly destroys brain cells, causing memory loss and cognitive decline over time.

Graph-structured data is ubiquitous in a wide range of real-world application domains, including social networks, biological protein-protein interaction networks, molecular graph structures, and brain connectivity networks. Graphs provide a flexible way to inherently represent real-world entities as a set of nodes and their interactions as a set of edges. A case in point: for brain analysis in populations and diagnosis, we model populations as graphs, where each node represents a subject with an associated node feature vector obtained from imaging data, and each edge represents a pairwise similarity between two subjects with an edge feature vector acquired from non-imaging data.

In recent years, there has been a surge of interest in extending deep learning approaches to non-Euclidean domains thanks, in large part, to the prevalence and increasing proliferation of graph-structured data [21,23,24,84]. Advances in deep learning have spawned significant efforts to facilitate, for instance, the clinical diagnosis of brain diseases. Graph convolutional networks (GCNs), which generalize convolutional neural networks to graph-structured data by leveraging spectral graph theory and its extensions, have gained popularity in graph representation learning [21] for their ability to capture the graph structure. GCN uses a layer-wise propagation rule based on a first-order approximation of spectral graph convolutions, where the feature vector of each graph node is updated by applying a weighted sum of the features of its immediate neighboring nodes. Wu *et al.* [84] introduce a simple graph convolution by removing the nonlinear transition functions between the layers of graph convolutional networks and collapsing the resulting function into a single linear transformation via the powers of the normalized adjacency matrix with added self-loops for all graph nodes. However, this simple graph convolution acts as a low-pass filter, which attenuates all but the zero frequency, causing oversmoothing. Zeng *et al.* [23] propose a graph sampling based learning method by sampling the training graph in lieu of nodes or edges across GCN layers, as well as eliminating biases in minibatch estimation via aggregator normalization techniques. Chen *et al.* [24] present an extension of the GCN model using skip connections from the input layer and identity mapping with the learnable weight matrix of each layer in a bid to alleviate the oversmoothing problem, which is a common effect of increasing the network depth.

The primary objective of graph convolution based methods is to learn node representations that encode structural information about the graph. These learned node representations can then be used as input to machine learning models for downstream tasks such as node classification whose goal is to predict the most probable labels of nodes in a graph. For instance, in brain diagnosis

tasks, which is the focus of this work, we want to classify subjects as diseased or healthy by predicting the node labels in a population graph. Graph convolution based methods have recently become prevalent in the biomedical and medical imaging domains [85–89] due largely to the fact that neuroimaging provides valuable information about the diagnosis and progression of brain diseases. Built on top of graph signal processing approaches [90], GCNs have shown promising results in metric learning and classification tasks on brain connectivity networks [91, 92]. Ktena *et al.* [91] propose to learn a graph similarity metric using a siamese graph convolutional neural network in a supervised fashion, yielding encouraging results in individual subject classification and manifold learning tasks. Similarly, Ma *et al.* [92] introduce a higher-order siamese graph convolutional neural network for multi-subject brain analysis in health and neuropsychiatric disorders by incorporating higher-order proximity in graph convolutions, with the goal of characterizing the community structure of brain connectivity networks and learning the similarity among magnetic resonance imaging (fMRI) brain connectivity networks extracted from multiple subjects. While GCNs have also been successfully used in the prediction of developmental and brain disorders such as autism spectrum disorder (ASD) and Alzheimer's disease (AD) [4, 27, 28], they are prone to the oversmoothing problem, where learned node representations become similar due to repeated graph convolutions as the network depth increases. In other words, when the number of GCN layers increases, the learned node representations tend to converge to indistinguishable feature vectors, resulting in performance degradation and less expressiveness; and hence the model becomes less aware of the graph structure.

In order to overcome the aforementioned issues, we propose an aggregator normalization graph convolutional network (AN-GCN) with skip connections and identity mapping for the detection of neurodevelopmental and neurodegenerative brain disorders by integrating both imaging and non-imaging features into the graph nodes and edges, respectively. We formulate the disease prediction problem as a semi-supervised node classification on population graphs. The main contributions of this work can be summarized as follows:

- We propose a novel graph convolutional aggregation approach with skip connections and identity mapping for node classification by effectively integrating into the graph both imaging and non-imaging information.
- We employ an aggregator normalization mechanism for feature propagation in an effort to eliminate bias in minibatch estimation.
- We show through experimental results that our model yields competitive performance in comparison with strong baselines on two large benchmark datasets.

The remainder of this chapter is organized as follows. In Section 2, we review important relevant work. In Section 3, we present the problem formulation as a semi-supervised node classification task, and then we introduce a two-stage graph convolutional aggregation framework for disease prediction. In the first stage, we construct a population graph comprised of a node set and an edge set with complementary imaging and non-imaging data, respectively. Each graph node represents a subject with an associated feature vector extracted from imaging data, and each edge captures similarities between a pair of subjects with non-imaging data integrated into the edge weight. In the second stage, we design an aggregator normalization graph convolutional network architecture by leveraging skip connections, identity mapping and aggregation in graph sampling. In Section 4, we present experimental results to demonstrate the competitive performance of our approach in comparison with graph-based methods for brain disease prediction. Finally, we conclude in Section 5 and highlight some promising directions for future work.

4.2 Related Work

The basic objective of node classification in populations and diagnosis is to predict the most probable labels of nodes in a population graph, where each subject is represented by a node and each edge encodes the pairwise similarity between a pair of connected nodes. To achieve this objective, various graph convolution based methods have been proposed with the aim of distinguishing between diseased patients and healthy controls by predicting the node labels (i.e. clinical status of subjects). In semi-supervised node classification, the amount of labeled nodes for model training is typically small and the goal is to predict the labels of a large number of unlabeled nodes by learning a prediction rule from both labeled and unlabeled nodes in order to improve model performance.

Graph Convolutional Networks. GCNs have recently become the model of choice in semi-supervised node classification tasks [21]. GCN uses an efficient layer-wise propagation rule, which is based on a first-order approximation of spectral graph convolutions. The feature vector of each graph node is updated by essentially applying a weighted sum of the features of its neighboring nodes. Xu *et al.* [22] introduce a graph wavelet neural network, which is a GCN-based architecture that uses spectral graph wavelets in lieu of graph Fourier bases to define a graph convolution. Despite the fact that spectral graph wavelets can yield localization of graph signals in both spatial and spectral domains, they require explicit computation of the Laplacian eigenbasis, leading to a high computational complexity, especially for large graphs.

While GCNs have shown great promise, achieving state-of-the-art performance in semi-supervised node classification tasks, they are prone to oversmoothing the node features. In fact,

the neighborhood aggregation scheme (i.e. graph convolution) of GCN is tantamount to applying Laplacian smoothing [93], which replaces each graph node with the average of its immediate neighbors. Therefore, repeated application of GCN yields smoother and smoother versions of the initial node features as the number of the network's layers increases. As a result, the node features in deeper layers will eventually converge to the same value, and hence become too similar across different classes. Wu *et al.* [84] introduce a simple graph convolution by removing the nonlinear transition functions between the layers of graph convolutional networks and collapsing the resulting function into a single linear transformation via the powers of the normalized adjacency matrix with added self-loops for all graph nodes. However, this simple graph convolution acts as a low-pass filter, which attenuates all but the zero frequency, causing oversmoothing. Significant strides have been made toward remedying the problem of oversmoothing in GCNs [24, 94, 95]. Xu *et al.* [94] propose jumping knowledge networks, which employ dense skip connections to connect each layer of the network with the last layer to preserve the locality of node representations in order circumvent oversmoothing. In [95], a normalization layer, which helps avoid oversmoothing by preventing learned representations of distant nodes from becoming indistinguishable, has been proposed. This normalization layer is performed on intermediate layers during training, and the aim is to apply smoothing over nodes within the same cluster while avoiding smoothing over nodes from different clusters. Chen *et al.* [24] design a deep graph convolutional with initial residual and identity mapping to tackle the problem of oversmoothing by adding an identity matrix to the learnable weight matrix and skip connections from the initial feature matrix.

Disease Prediction. GCNs have recently shown great potential in neuroimaging and computer aided diagnosis, especially in the prediction of brain diseases such as autism spectrum disorder and Alzheimer's disease [4, 25–30]. Using a graph convolutional neural network model consisting of a fully convolutional GCN with several hidden layers activated via the Rectified Linear Unit function, Parisot *et al.* [4] introduce a disease prediction framework. It involves modeling a population as a graph with nodes representing subjects and edges encoding the similarity between a pair of subjects by combining imaging and non-imaging information in order to improve model classification performance with the goal of distinguishing between patients with autism spectrum disorder and healthy controls, as well as predicting whether a patient with mild cognitive impairment will convert to Alzheimer's disease. The graph nodes are associated with imaging-based features, while non-imaging data is integrated into the edge weights. To learn an adaptive graph representation for GCN learning, Zheng *et al.* [28] integrate graph learning and graph convolution to develop an end-to-end multimodal graph learning approach for disease prediction via a multimodal fusion module, which fuses the features of each modality by leveraging the correlation and

complementarity between the modalities. Cao *et al.* [27] introduce a deep learning model using a multi-layer GCN in conjunction with residual neural networks to tackle the vanishing gradient problem, and the DropEdge technique [96] to alleviate overfitting and oversmoothing, which are two major challenges in developing deep GCNs for node classification. Similar to Dropout technique that randomly sets the outgoing edges of hidden units to zero at each update of the training phase, DropEdge can be regarded as an extension of Dropout to graph edges. Inspired by the Inception network in convolutional neural networks, Kazi *et al.* [25] propose an Inception graph convolutional network for disease prediction tasks with complementary imaging and non-imaging multi-modal data by leveraging spectral convolutions with different kernel sizes, showing improved performance over regular GCN architectures. Cosmo *et al.* [26] present an end-to-end trainable graph learning architecture for dynamic and localized graph pruning with the aim to build a node classification model consisting of few graph convolutional layers, followed by a fully connected layer to predict the patient label. Building upon GCNs, Jiang *et al.* [97] introduce a hierarchical GCN model for graph embedding learning of brain network and brain disorders prediction by hierarchically learning deep representations from functional fMRI brain connectivity networks in order to improve classification performance for disease diagnosis. Pan *et al.* [29] propose a diagnosis classification framework that incorporates self-attention graph pooling and graph convolutional networks by extracting features from the non-Euclidean brain network, as well as fusing both imaging and non-imaging information with the aim to detect inter-group heterogeneity and intra-group homogeneity regarding brain activities. While these approaches have shown promising results in brain disease prediction tasks, they are, however, prone to the issue of oversmoothing.

4.3 Method

In this section, we introduce our notation and formulate the disease prediction problem as a semi-supervised node classification task on population graphs, which are used to model pairwise relations (edges) between subjects (nodes). Each graph node and edge weight are associated with complementary imaging and non-imaging data, respectively. Then, we present the main building blocks of the proposed network architecture for graph representation learning and semi-supervised node classification.

4.3.1 Preliminaries and Problem Statement

Basic Notions. Let $\mathbb{G} = (\mathcal{V}, \mathcal{E})$ be a graph, where $\mathcal{V} = \{1, \ldots, N\}$ is the set of N nodes and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of edges. We denote by $\mathbf{A} = (\mathbf{A}_{ij})$ an $N \times N$ adjacency matrix (binary

or real-valued) whose (i, j)-th entry $\mathbf{A}_{ij}$ is equal to the weight of the edge between neighboring nodes i and j, and 0 otherwise. We also denote by $\mathbf{X} = (\mathbf{x}_1, ..., \mathbf{x}_N)^\intercal$ an $N \times F$ feature matrix of node attributes, where $\mathbf{x}_i$ is an F-dimensional row vector for node i.

Learning latent representations of nodes in a graph aims at encoding the graph structure into low-dimensional embeddings, such that both structural and semantic information are captured. More precisely, the purpose of network/graph embedding is to learn a mapping $\varphi : \mathcal{V} \rightarrow \mathbb{R}^P$ that maps each node i to a P-dimensional vector $\mathbf{z}_i$, where $P \ll N$. These learned node embeddings can then be used as input to learning algorithms for downstream tasks, such as node classification.

Problem Statement. Given the labels of a subset of the graph nodes (or their corresponding final output embeddings), the objective of semi-supervised learning is to predict the unknown labels of the other nodes. More specifically, let $\mathcal{D}_l = \{(\mathbf{z}_i, y_i)\}_{i=1}^{N_l}$ be the set of labeled final output node embeddings $\mathbf{z}_i \in \mathbb{R}^P$ with associated known labels $y_i \in \mathcal{Y}_l$, and $\mathcal{D}_u = \{\mathbf{z}_i\}_{i=N_l+1}^{N_l+N_u}$ be the set of unlabeled final output node embeddings, where $N_l + N_u = N$. Then, the problem of semi-supervised node classification is to learn a classifier $f : \mathcal{V} \rightarrow \mathcal{Y}_l$. That is, the goal is to predict the labels of the set $\mathcal{D}_u$.

It is important to note that for multi-class classification problems, the label of each node i (or its final output embedding $\mathbf{z}_i$) in the labeled set $\mathcal{D}_l$ can be represented as a C-dimensional one-hot vector $\mathbf{y}_i \in \{0, 1\}^C$, where C is the number of classes with 0 and 1 representing "healthy" and "diseased" status of the subjects, respectively.

4.3.2 Proposed Model

We now describe our proposed model, a two-stage approach for graph representation learning and semi-supervised node classification. The aim is to learn discriminative node embeddings for computer aided diagnosis. In the first stage, we construct a population graph, which is a vital step in designing a GCN-based prediction model since GCNs rely on the affinity matrix between subjects to update their layer-wise feature propagation rules. Hence, to fully exploit the expressive power of GCNs, an appropriately constructed graph that accurately explains the similarity between subjects is of paramount importance in graph representation learning, especially in computer aided diagnosis tasks. In the second stage, we introduce a disease prediction model by leveraging graph convolutional aggregation in conjunction with skip connections and identity mapping.

Population Graph Construction

Following the population graph construction in graph convolutional networks for disease prediction [4], we also combine both imaging and non-imaging data in our proposed approach. More specifically, we model a population as a graph consisting of nodes representing subjects and edges capturing pairwise similarities between subjects. Each node has a feature vector extracted from imaging data, whereas each edge weight represents phenotypic (i.e. non-imaging) data. The graph construction is shown in Figure 4.1, where the nodes are associated with imaging-based feature vectors, while phenotypic (non-imaging) information is incorporated as edge weights.

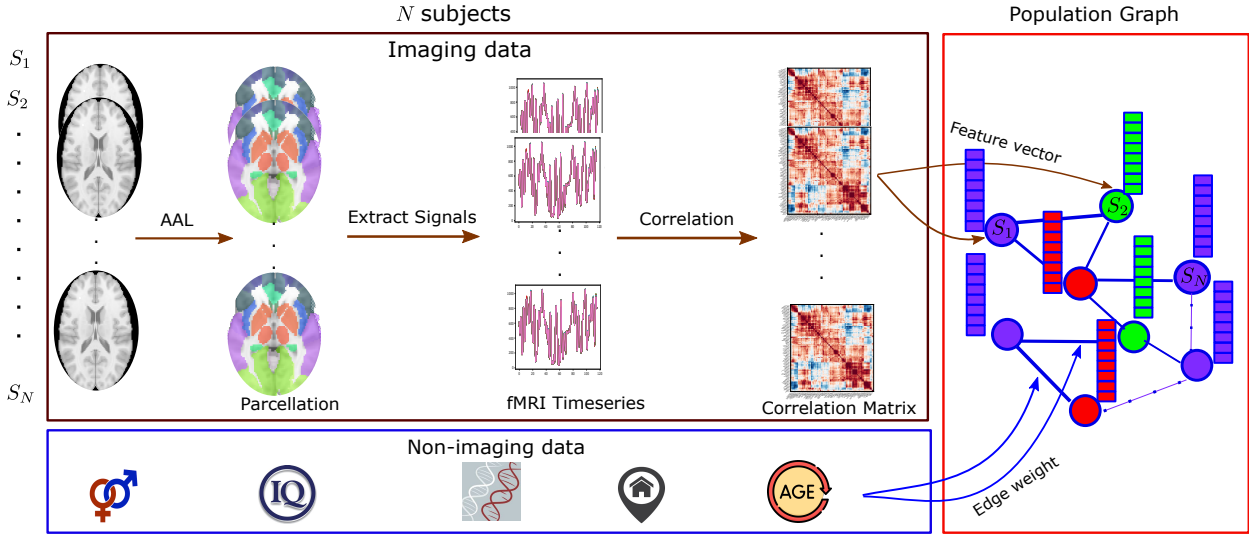

Figure 4.1: Graph construction from N subjects using imaging and non-imaging data. For imaging data, we employ Automated Anatomical Labeling (AAL) to perform brain parcellation.

Let $\{M_1, \ldots, M_T\}$ be a set of T non-imaging phenotypic measures such as a subject's age or gender. The adjacency matrix $\mathbf{A} = (\mathbf{A}_{ij})$ of a population graph comprised of N subjects is defined as

$$\mathbf{A}_{ij} = K(i,j) \sum_{t=1}^{T} d(M_t(i), M_t(j)), \tag{4.1}$$

where $K(i,j) = \text{similarity}(S_i, S_j)$ denotes a kernel similarity between subjects S_i and S_j (i.e. edge weight between graph nodes i and j), and d is a pairwise distance between phenotypic measures. The kernel similarity measure $K(i,j)$ is given by

$$K(i,j) = \exp\Big(- \frac{\rho(\mathbf{x}_i, \mathbf{x}_j)^2}{2\sigma^2} \Big), \tag{4.2}$$

where σ is a smoothing parameter, which determines the width of the kernel, and ρ is the correlation

distance between feature vectors $\mathbf{x}_i$ and $\mathbf{x}_j$ for nodes i and j, respectively,

$$\rho(\mathbf{x}_i, \mathbf{x}_j) = 1 - \frac{(\mathbf{x}_i - \bar{\mathbf{x}}_i)(\mathbf{x}_j - \bar{\mathbf{x}}_j)^\intercal}{\|\mathbf{x}_i - \bar{\mathbf{x}}_i\|\|\mathbf{x}_j - \bar{\mathbf{x}}_j\|}, \tag{4.3}$$

with $\bar{\mathbf{x}}_i = (\mathbf{x}_i\mathbf{1}/N)\mathbf{1}^\intercal$ and $\bar{\mathbf{x}}_j = (\mathbf{x}_j\mathbf{1}/N)\mathbf{1}^\intercal$ denoting row vectors whose elements are all equal to the mean of the components of $\mathbf{x}_i$ and $\mathbf{x}_j$, respectively, and $\mathbf{1}$ is an N-dimensional column vector of all ones.

The pairwise distance between phenotypic measures is defined depending on the kind of phenotypic data incorporated in the graph. Most phenotypic data can be classified into two main categories: qualitative (e.g. subject's gender) and quantitative (e.g. subject's age). For qualitative data, the distance measure is defined as

$$d(M_t(i), M_t(j)) = \begin{cases} 1 & \text{if } M_t(i) = M_t(j) \\ 0 & \text{otherwise.} \end{cases} \tag{4.4}$$

and for quantitative data, it is given by

$$d(M_t(i), M_t(j)) = \begin{cases} 1 & \text{if } |M_t(i) - M_t(j)| < \tau, \\ 0 & \text{otherwise.} \end{cases} \tag{4.5}$$

where τ is a given threshold.

Disease Prediction Model

Graph convolutional networks learn a new feature representation for each node such that nodes with the same labels have similar features [21].

Feature Diffusion. We denote by $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$ the adjacency matrix with self-added loops, where $\mathbf{I}$ is the identity matrix. The layer-wise feature diffusion rule of an L-layer GCN is given by

$$\mathbf{S}^{(\ell)} = \hat{\mathbf{A}}\mathbf{H}^{(\ell)}, \quad \ell = 0, \ldots, L-1, \tag{4.6}$$

where $\hat{\mathbf{A}} = \tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}$ is the normalized adjacency matrix with self-added loops, $\tilde{\mathbf{D}} = \mathrm{diag}(\tilde{\mathbf{A}}\mathbf{1})$ is the diagonal degree matrix, and $\mathbf{H}^{(\ell)} \in \mathbb{R}^{N \times F_\ell}$ is the input feature matrix of the ℓ-th layer with F_ℓ feature maps. The input of the first layer is the original feature matrix $\mathbf{H}^{(0)} = \mathbf{X}$.

Aggregated Feature Diffusion. Inspired by the aggregation mechanism in graph sampling [23], we define a layer-wise aggregated feature diffusion rule for node features in the ℓ-th layer as follows:

$$\mathbf{S}^{(\ell)} = (\hat{\mathbf{A}} \odot \mathbf{\Gamma})\mathbf{H}^{(\ell)}, \tag{4.7}$$

where $\odot$ denote element-wise matrix multiplication, and $\mathbf{\Gamma} = (\gamma_{ij})$ is an $N \times N$ aggregation matrix. Each element γ_{ij} is an aggregator normalization constant given by

$$\gamma_{ij} = \frac{C_i}{C_{ij}}, \tag{4.8}$$

where C_i and C_{ij} denote the number of times the node $i \in \mathcal{V}$ or edge $(i, j) \in \mathcal{E}$ appears in the subgraphs of $\mathbb{G} = (\mathcal{V}, \mathcal{E})$, respectively. These subgraphs are obtained by running the GraphSaint sampler repeatedly before the training starts [23].

Learning Node Embeddings. Motivated by the good performance of graph sampling and identity mapping in alleviating the oversmoothing problem in graph representation learning [23, 24, 84, 98, 99], we propose an aggregator normalization graph convolutional network (AN-GCN) by leveraging aggregation in graph sampling, as well as skip connections and identity mapping. The output feature matrix $\mathbf{H}^{(\ell+1)}$ of our proposed AN-GCN model is obtained by applying the following layer-wise propagation rule

$$\begin{aligned} \mathbf{H}^{(\ell+1)} = \sigma\Bigg(& \big(1 - \alpha_\ell\big)(\hat{\mathbf{A}} \odot \mathbf{\Gamma})\mathbf{H}^{(\ell)} \\ & + \beta_\ell(\hat{\mathbf{A}} \odot \mathbf{\Gamma})\mathbf{H}^{(\ell)}\big(\mathbf{I} + \mathbf{W}^{(\ell)}\big) \\ & + \alpha_\ell \mathbf{X} + \beta_\ell \mathbf{X}\big(\mathbf{I} + \mathbf{W}^{(\ell)}\big)\Bigg), \end{aligned} \tag{4.9}$$

where α_ℓ and β_ℓ are nonnegative hyper-parameters in the interval $[0, 1]$ and are often fine-tuned via grid search, and $\sigma(.)$ is a point-wise non-linear activation function such as $\text{ReLU}(.) = \max(0, .)$. The use of skip connections ensures that the final representation of each node retains at least a percentage α_ℓ of feature data from the input layer, while identity mapping not only imposes regularization on the learnable weight matrix in order to avoid over-fitting, but it is also beneficial to semi-supervised learning tasks where training data is limited [100].

Model Prediction. The embedding matrix $\mathbf{Z} = \mathbf{H}^{(L)}$ of the last layer of AN-GCN contains the final output node embeddings, and captures the neighborhood structural information of the graph within L hops. This final node representation can be used as input for node classification. To this end, we apply a softmax classifier as follows:

$$\hat{\mathbf{Y}} = \text{softmax}(\mathbf{Z}), \tag{4.10}$$

where $\hat{\mathbf{Y}} \in \mathbb{R}^{N \times C}$ is the matrix of predicted labels for graph nodes, and C is the total number of classes. The softmax classifier is a generalization of the binary logistic regression classifier to multiple classes, and as the name suggests it uses the softmax function that turns a vector of C

real-valued class scores into a vector of C normalized positive scores that sum to 1. In other words, the softmax classifier returns probability scores for all classes.

Model Training. For semi-supervised multi-class classification, the neural network weight parameters are learned by minimizing the cross-entropy loss function

$$\mathcal{L} = -\sum_{i \in \mathcal{Y}_l} \sum_{c=1}^{C} \mathbf{Y}_{ic} \log \hat{\mathbf{Y}}_{ic}, \tag{4.11}$$

over the set $\mathcal{Y}_l$ of all labeled nodes, where $\mathbf{Y}_{ic}$ is equal 1 if node i belongs to class c, and 0 otherwise; and $\hat{\mathbf{Y}}_{ic}$ is the (i, c)-element of the matrix $\hat{\mathbf{Y}}$ from the softmax function, i.e. the probability that the network associates the i-th node with class c. During training, the network parameters are updated using the Adam optimizer [101].

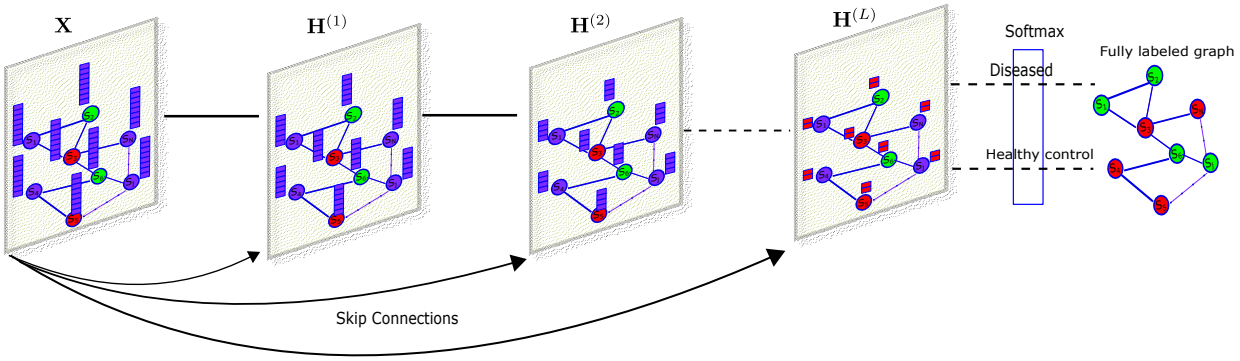

Figure 4.2: Schematic layout of the proposed AN-GCN architecture.

4.4 Experiments

In this section, we conduct several experiments to assess the performance of the proposed AN-GCN model on two standard datasets for disease prediction. More specifically, we address the disease prediction problem as a semi-supervised node classification task, and the goal is predict the label (i.e. clinical status) of a test node (i.e. subject) in a population graph as diseased or healthy, where only a small number of nodes are labeled. The effectiveness of our model is validated through experimental comparison with strong baseline methods. While presenting and analyzing our experimental results, we aim to answer the following main research questions (RQs):

- **RQ1:** How does AN-GCN perform in comparison with state-of-the-art disease prediction models?
- **RQ2:** How does AN-GCN alleviate the oversmoothing problem?
- **RQ3:** What is the effect of hyperparameters on the performance of AN-GCN?

4.4.1 Experimental Setup

Datasets

We evaluate the proposed model on two large datasets, namely ABIDE and ADNI.

- **ABIDE Dataset:** The Autism Brain Imaging Data Exchange (ABIDE)[1] initiative aggregates resting-state functional magnetic resonance imaging (rs-fMRI) and phenotypic data of 1112 subjects from various international brain imaging laboratories. We select a set of 871 subjects, consisting of 403 ASD patients and 468 healthy controls (HCs). As a result of the different acquisition sites, the ABIDE dataset is heterogeneous, and the aim is to separate ASD subjects from healthy controls.

- **ADNI Dataset:** The Alzheimer's Disease Neuroimaging Initiative (ADNI)[2] is a North American multisite study designed to develop clinical, neuroimaging techniques, biochemical and genetic biomarkers for the early detection and tracking of patients with Alzheimer's disease (AD), as well as subjects with mild AD, normal subjects, and subjects with mild cognitive impairment (MCI). ADNI has recruited more than 1700 adults, aged 55 to 90 years, from over 50 sites across the United States and Canada for its four studies (ADNI-1, 2, 3 and -GO). We select a set of 573 participants, comprised of 402 HC individuals and 171 MCI subjects. The aim is to predict whether an MCI subject will convert to AD.

Data Preprocessing

For fair comparison, we follow the same data preprocessing procedure laid out in the GCN baseline [4]. For preprocessing of the ABIDE dataset, we use the Configurable Pipeline for the Analysis of Connectomes (C-PAC), which includes skull stripping, slice timing correction, motion correction, global mean intensity normalization, nuisance signal regression, band-pass filtering (0.01-0.1Hz), and registration of fMRI images to a standard anatomical space. Then, the mean timeseries for a set of cortical and subcortical regions of interest (ROIs) extracted from the Harvard Oxford atlas are computed and standardized using z-score normalization to ensure the timeseries distributions have mean zero mean and unit variance. The goal of z-score normalization is to transform timeseries to be on a similar scale in an effort to improve the performance and training stability of the model. Subsequently, we compute N connectivity matrices using the Pearson's correlation coefficient between the representative rs-fMRI timeseries of each ROI in the Harvard Oxford atlas. Since z-scores are not necessarily normally distributed, we apply Fisher z-transformation,

[1] http://preprocessed-connectomes-project.org/abide/
[2] http://adni.loni.usc.edu/

which is the inverse hyperbolic tangent function that converts Pearson's correlation coefficient to a normally distributed variable. In other words, the correlation matrices are Fisher transformed in order to convert the skewed distribution of the correlation coefficient into a distribution that is approximately normal. It is also worth pointing out that the variance of the Fisher transformed distribution is independent of the correlation, whereas the variance of the sampling distribution of the correlation coefficient depends on the correlation. For the edge weights of the population graph on the ABIDE datset, we incorporate the subject's gender, age and acquisition site as as phenotypic measures. For the ADNI dataset, we parcellate each 3D brain volume into N ROIs using Automated Anatomical Labeling (AAL), followed by computing the connectivity matrices between timeseries. The edge weights of population graph on the ADNI dataset consist of the subject's gender and age as phenotypic measures. Since the correlation matrix is symmetric, it suffices to use either its upper or lower triangular part. Hence, we take the upper triangular part and vectorize it to obtain a feature vector whose dimension is then further reduced using recursive feature elimination via a ridge classifier.

Performance Evaluation Metrics

A classification model's performance is normally evaluated by applying it to test data with known target values and comparing the predicted values to the known values. We use Accuracy (Acc), Area Under Curve (AUC), Recall, Precision, F1 score, Matthews Correlation Coefficient (MCC), and Cohen's kappa (κ) as evaluation metrics, which are defined as

$$\text{Acc} = \frac{\text{TP + TN}}{\text{TP + TN + FP + FN}},$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP + FP}},$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP + FN}},$$

$$\text{F1} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}},$$

$$\text{MCC} = \frac{\text{TP} \times \text{TN} - \text{FP} \times \text{FN}}{\sqrt{(\text{TP + FP})(\text{TN + FP})(\text{TP + FN})(\text{TN + FN})}},$$

and

$$\kappa = \frac{2 \times (\text{TP} \times \text{TN} - \text{FP} \times \text{FN})}{(\text{TP} + \text{FP}) \times (\text{TN} + \text{FP}) + (\text{TP} + \text{FN}) \times (\text{TN} + \text{FN})},$$

where TP, FP, TN and FN denote number of true positives, false positives, true negatives and false negatives, respectively.

The F1-score is defined as the harmonic mean of precision and recall. The harmonic mean is more intuitive than the arithmetic mean when computing a mean of ratios. The F1-score will only be high if both precision and recall have high values. This is due to the fact that the harmonic mean of two numbers is always closer to their minimum.

We also use AUC, the area under the receiving operating characteristic (ROC) curve, as a metric. AUC summarizes the information contained in the ROC curve, which plots the true positive rate versus the false positive rate, at various thresholds. Larger AUC values indicate better performance at distinguishing between healthy and diseased subjects. An uninformative classifier has an AUC equal to 50% or less. An AUC of 50% corresponds to a random classifier (i.e. for every correct prediction, the next prediction will be incorrect), whereas an AUC score smaller than 50% indicates that the classifier performs worse than a random one.

Baseline Methods

We evaluate the performance of the proposed AN-DGCN model against various graph convolutional based methods for computer aided diagnosis, including GCN for disease prediction [4], multi-modal graph learning (MMGL) for disease prediction [28], DeepGCN for autism spectrum disorder identification from multi-site resting-state data [27], InceptionGCN for disease prediction [25], latent-graph learning (LGL) for disease prediction [26], edge-variational graph convolutional network (EV-GCN) for uncertainty-aware disease prediction [102], down-sampling and multi-modal learning (DS-MML) for identifying autism spectrum disorder [29], hierarchical graph convolution network (HI-GCN) for brain disorders prediction [97], brain connectivity via graph convolution network (BN-GCN) for Alzheimer's Disease [103], and mutual multi-scale triplet graph convolutional network (MMTGCN) for brain disorders classification [30]. We also compare our model to logistic regression, gradient boosting, and tensor-train, high-order pooling and semi-supervised learning-based generative adversarial network (THS-GAN) [104].

Implementation Details

All experiments are carried out on a Linux workstation running Intel(R) CPU 2.40 GHz and 128-GB RAM with an V100-SXM2 16-GB GPU. The proposed model is implemented in PyTorch and trained for 150 and 100 epochs on the ABIDE and ADNI datasets, respectively, using Adam optimizer with a learning rate of 10^{-3}. The values of the hyperparameters α_ℓ and β_ℓ are set to 0.1 and 0.3 for the ABIDE dataset, and 0.1 and 0.2 for the ADNI dataset, respectively, via grid search with cross-validation on the training set. We use a stratified 10-fold cross-validation strategy. We also set the number of layers for our model to $L = 10$. The training is stopped when the validation

loss does not decrease after 10 consecutive epochs. The values of the cross-entropy metric are recorded at the end of each epoch on the training set. The performance comparison plots of AN-GCN and GCN over training epochs on the training set of the ABIDE dataset are visualized in Figure 4.3, which shows that both models yield comparable training loss values. However, as the number of epochs increases, our model yields lower training loss values, indicating better predictive accuracy.

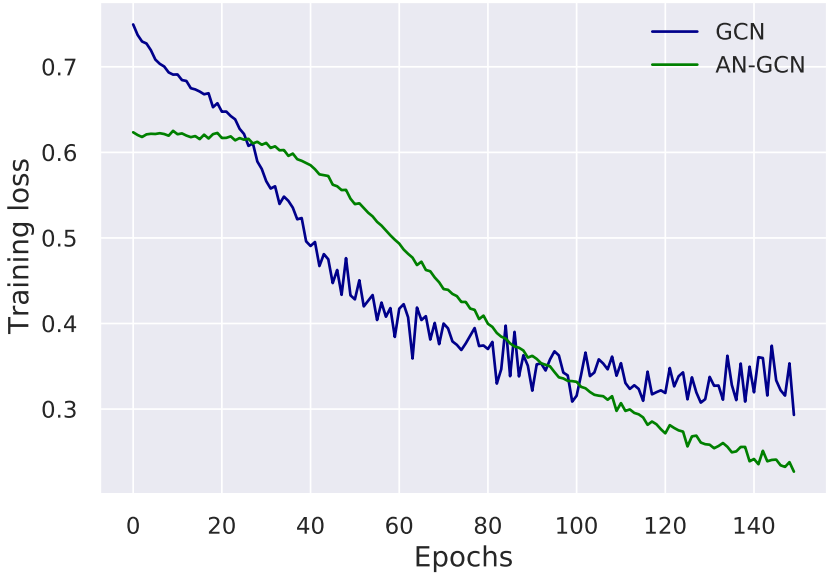

Figure 4.3: Model training history comparison between GCN and our AN-GCN model on the ABIDE dataset.

4.4.2 Experimental Results and Analysis

In order to answer **RQ1**, we report the classification performance of AN-GCN and baseline methods in Table 4.1 using seven evaluation metrics, including average accuracy, AUC and F1-score. Each metric is averaged across all test samples. As can be seen, the results show that our model outperforms all the baseline methods on the ABIDE dataset, achieving relative improvements of 50.33%, 34.10%, 50.33%, 12.25% and 50.33% over GCN in terms of accuracy, AUC, F1-score, recall and precision, respectively. The relative improvements over GCN are significant in terms of κ and MCC. Compared to the strongest baseline, our model outperforms DS-MML by relative improvements of 10.60%, 5.28% and 11.70% in terms of accuracy, AUC and recall, respectively.

Similarly, we report the performance comparison results of our model and baseline methods on the ADNI dataset in Table 4.2, which also shows that AN-GCN performs better than all the competing baselines. Our model yields relative improvements of 15.61%, 8.66%, 14.13% and 15.70% over GCN in terms of accuracy, AUC, F1-score and precision, respectively. Moreover,

Table 4.1: Performance comparison of our model and baseline methods on the ABIDE dataset using various evaluation metrics (%). Boldface numbers indicate the best classification performance.

Method	Accuracy	AUC	F1-score	Recall	Precision	κ	MCC
Logistic Regression	61.03	68.05	70.19	88.42	58.4	19.85	25.13
Gradient Boosting	59.97	62.04	62.24	63.48	61.34	19.53	19.67
GCN [4]	64.63	72.23	64.63	86.33	64.63	26.64	30.16
InceptionGCN [25]	72.69	72.81	79.27	-	-	-	-
EV-GCN [102]	80.83	84.98	81.24	-	-	-	-
LGL [26]	84.69	84.46	-	-	-	-	-
HI-GCN [97]	66.50	72.10	-	65.30	-	-	-
MMGL [28]	86.95	86.84	-	-	-	-	-
DeepGCN [27]	73.71	75.20	69.68	-	-	-	-
DS-MML [29]	87.62	92.00	-	86.76		-	-
AN-GCN (Ours)	**96.91**	**96.86**	**97.16**	**96.91**	**97.00**	**93.76**	**93.86**

AN-GCN significantly outperforms GCN in terms of recall, κ and MCC. In addition, our model outperforms the strongest baseline (i.e. BCN-GCN) by relative improvements of 5.76% and 4.36% in terms of accuracy and AUC, respectively.

Table 4.2: Performance comparison of our model and baseline methods on the ADNI dataset using various evaluation metrics (%). Boldface numbers indicate the best classification performance.

Method	Accuracy	AUC	F1-score	Recall	Precision	κ	MCC
Logistic Regression	58.71	51.61	68.80	58.71	59.67	04.48	04.04
Gradient Boosting	65.21	68.57	65.21	71.83	65.21	29.52	29.95
GCN [4]	84.98	89.32	84.89	58.82	84.98	60.37	62.58
HI-GCN [97]	75.40	75.60	-	66.40	-	-	-
BCN-GCN [103]	92.90	93.00	-	-	-	-	-
MMTGCN [30]	86.00	90.30	-	86.90	-	-	-
THS-GAN [104]	85.71	85.35	87.27	88.89	85.71	-	-
AN-GCN (Ours)	**98.25**	**97.06**	**96.89**	**98.25**	**98.33**	**95.68**	**95.84**

In order to visually compare the performance of the proposed model to the baseline methods, we use box plots across all the folds on the ABIDE and ADNI dataset using accuracy and AUC as evaluation metrics, as shown in Figures 4.4 and 4.5. A box plot is a simple method for graphically depicting groups of numerical data through their quartiles, and it is commonly used to assess and compare the shape, central tendency, and variability of sample distributions, as well as to identify

outliers. The box and whiskers show how the data is spread out. On each box, the central line represents the median, and the bottom and top edges of the box indicate the first and third quartile, respectively. The whiskers extend from the edges of the box to the lower and upper inner fences to show the range of the data. The fences are defined in terms of the inter-quartile range, and any value that falls outside the fences in considered as an outlier.

Figure 4.4 shows that our model outperforms the competing baselines in terms of the accuracy and AUC metrics for all the 10-folds. The higher the accuracy and AUC scores, the better the model distinguishes between patients suffering from ASD and healthy controls. As can be seen in Figure 4.4, the distribution of our model has less variability than GCN, gradient boosting and logistic regression in autism spectrum disorder prediction tasks. For instance, the median accuracy score for AN-GCN on the ABIDE dataset indicates significant difference in performance between our model and the three baseline methods. In addition, the box for AN-GCN is short, meaning that the accuracy values consistently hover around the average accuracy. However, the boxes for three baselines are taller, implying variable accuracy and AUC values compared to AN-GCN. We can also observe that the whisker is longer on the lower end of the box for GCN, indicating the distribution of both accuracy and AUC values is negatively skewed. For our AN-GCN model, the whisker lengths are short and roughly of the same length, indicating lower standard deviation and data symmetry, respectively.

Similarly, the box plots shown in Figures 4.5 indicate that our AN-GCN model outperforms the three baseline methods on the ADNI dataset in terms of both accuracy and AUC metrics. Interestingly, the box plot for GCN exhibits an outlier for accuracy values, as well as longer whiskers for AUC values. In addition, the box for the logistic regression is much taller than the other methods, indicating high variability in accuracy and AUC values. Gradient boosting also exhibits an outlier for AUC values.

We also evaluate the performance of our model against competing baselines using the precision-recall (PR) and receiver operating characteristic (ROC) curves on both ABIDE and ADNI datasets. The PR curve summarizes the trade-off between the true positive rate and the positive predictive value for a predictive model using different probability thresholds. Precision is a measure of result relevancy, while recall is a measure of how many truly relevant results are returned. A high area under the PR curve represents both high recall and high precision, where high precision relates to a low false positive rate, and high recall relates to a low false negative rate. Moreover, a PR curve that is closer to the upper left indicates a better performance. On the other hand, the ROC curve summarizes the trade-off between the true positive rate and false positive rate for a predictive model using different probability thresholds. The area under the ROC curve (AUC) is a measure

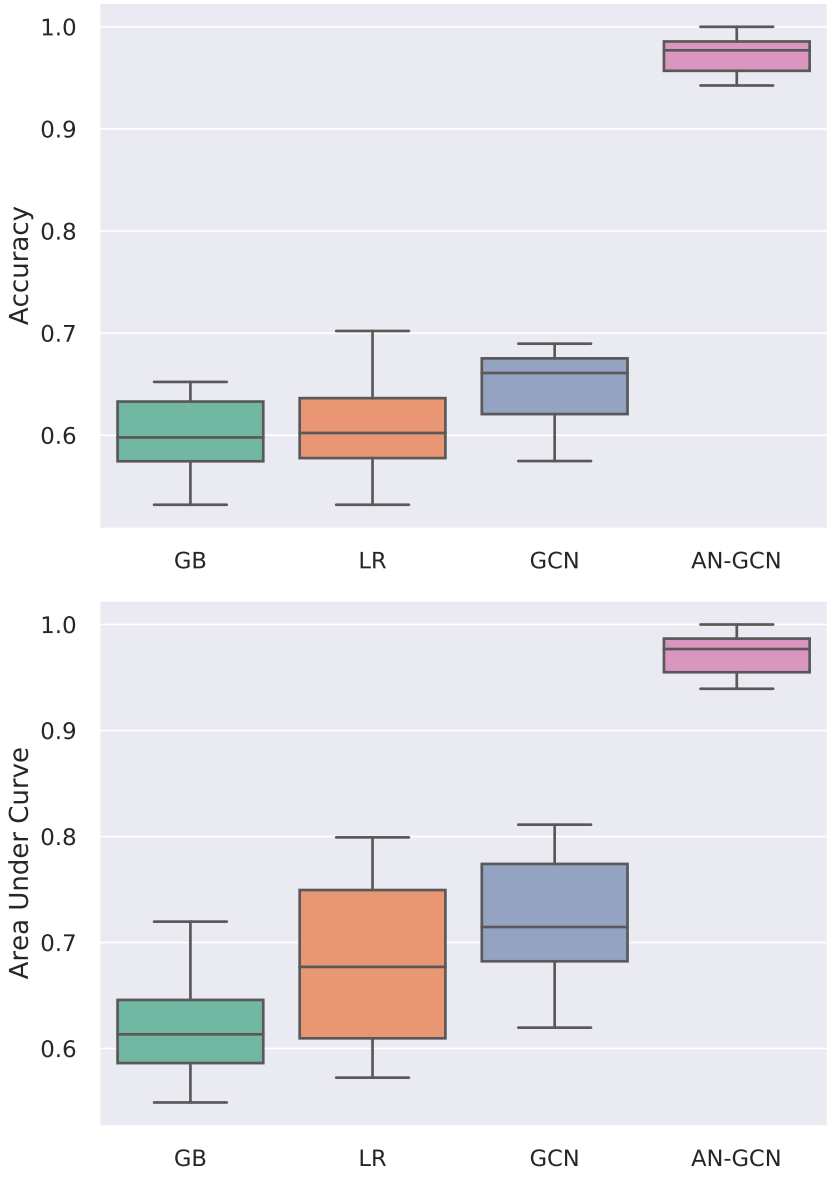

Figure 4.4: Comparative box plots of our model and baseline methods on the ABIDE dataset using accuracy and AUC scores over all cross-validation folds.

of discrimination in the sense that a model with a high AUC suggests that the model is able to accurately predict the value of an observation's response. Moreover, an ROC curve that is closer to the upper right indicates a better performance (i.e. true positive rate is higher than false positive rate).

Figures 4.6 and 4.7 show that our model yields the best performance compared to the baselines on both ABIDE and ADNI datasets. As can be seen, the PR (resp. ROC) curve of our model is much closer to the upper right (resp. left) than the corresponding curves for the baselines, indicating the better performance of AN-GCN in disease prediction tasks. In the ROC curves,

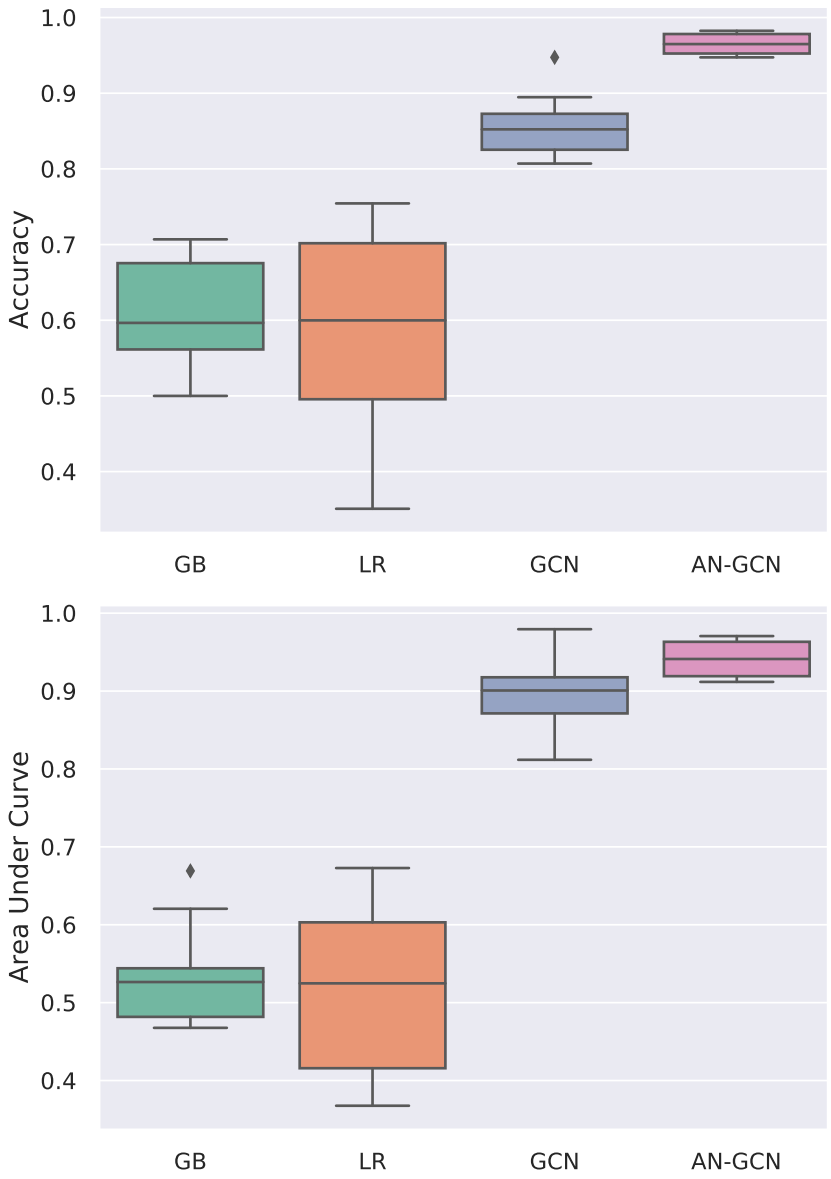

Figure 4.5: Comparative box plots of our model and baseline methods on the ADNI dataset using accuracy and AUC scores over all cross-validation folds.

the diagonal dashed line, which depicts a random algorithm (i.e., random guessing of classes), divides the ROC space. Points above the diagonal represent good classification results (better than random), points below the line poor results (worse than random). Notice that the ROC curves of the logistic regression and gradient boosting are closer to the diagonal line on the ABIDE dataset, indicating poor classification performance.

Overall, our AN-GCN model outperforms GCN and the other baselines significantly and consistently across all datasets, achieving state-of-the-art results in terms of various performance evaluation metrics. In particular, our model improves over the GCN baseline by a big margin. Another

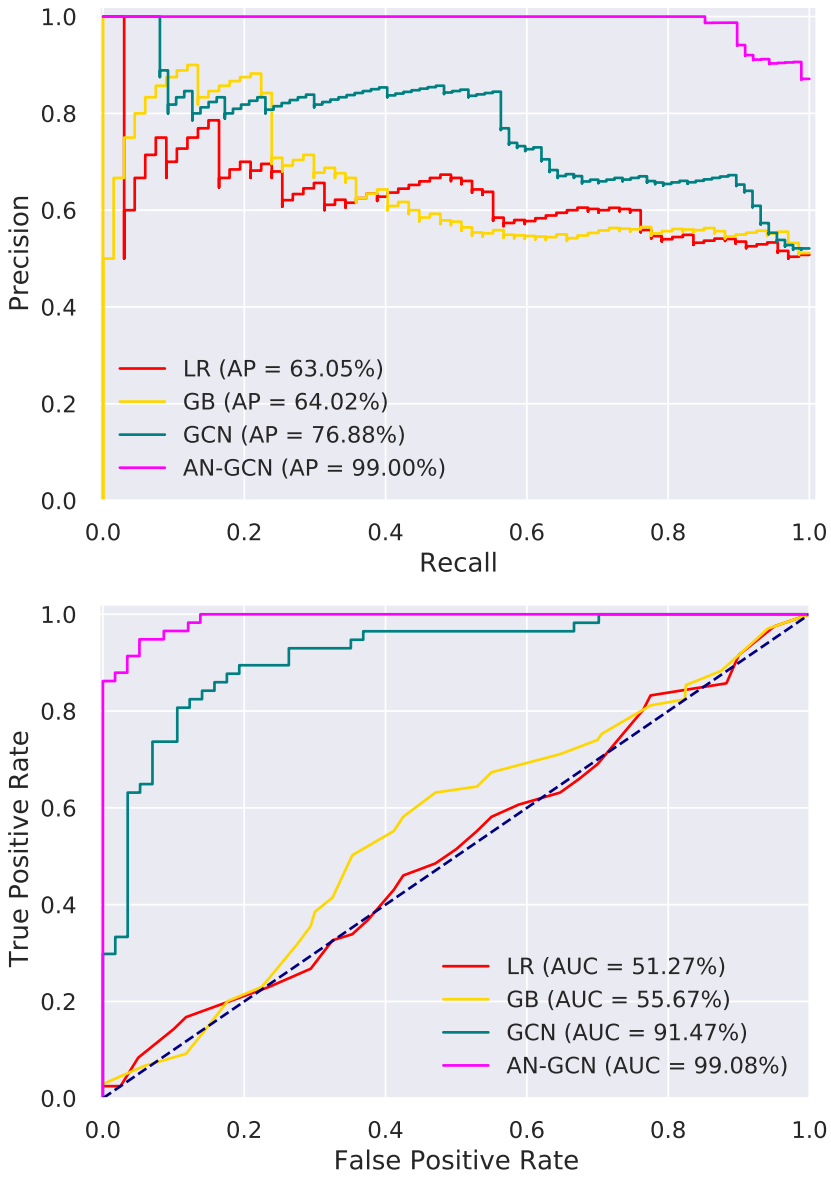

Figure 4.6: Precision-Recall and ROC curves of our model and baseline methods on the ABIDE dataset. Average precision (AP) and AUC values are enclosed in parentheses.

key observation is that AN-GCN also outperforms DeepGCN, yielding relative improvements of 31.47%, 28.80% and 39.44% over GCN in terms of accuracy, AUC and F1-score, respectively, on the ABIDE dataset.

4.4.3 Parameter Sensitivity Analysis

In order to answer **RQ2** and **RQ3**, we analyze the sensitivity of our disease prediction model to the choice of the number of network layers and the batch size. As the number of network layers

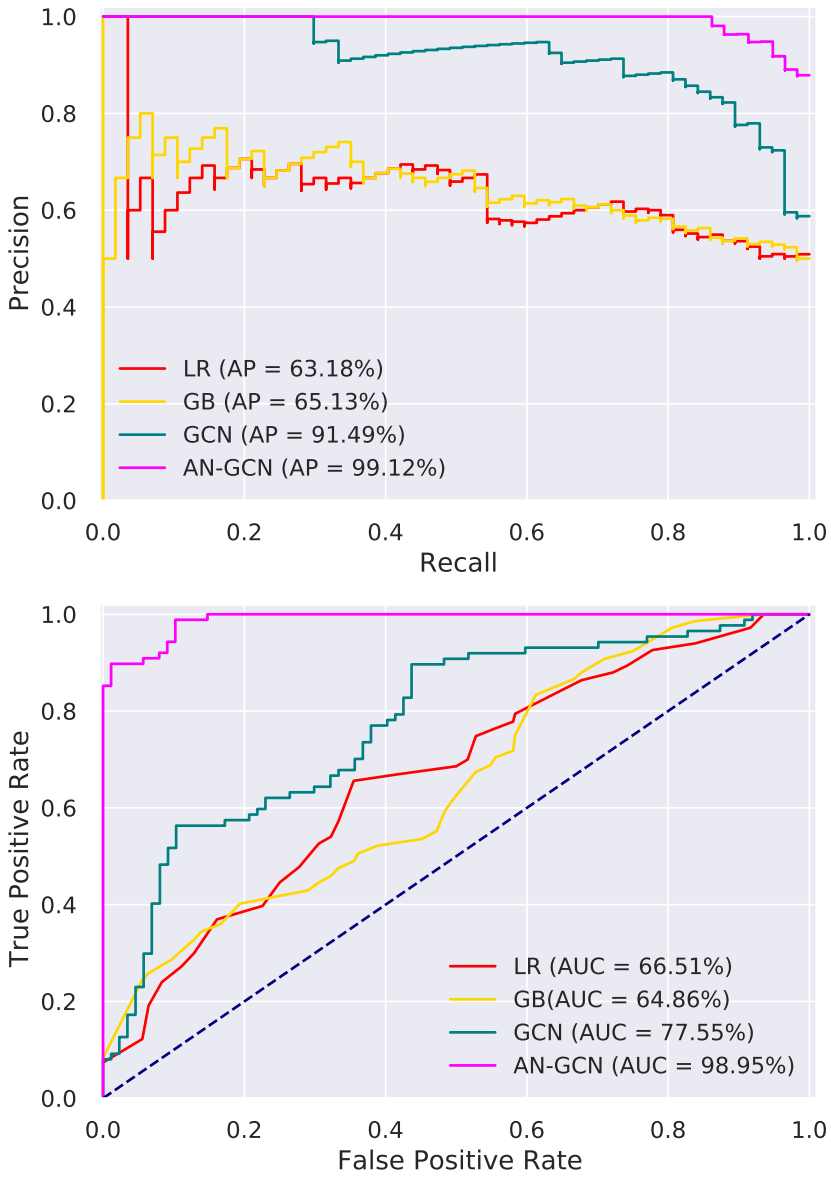

Figure 4.7: Precision-Recall and ROC curves of our model and baseline methods on the ADNI dataset. Average precision (AP) and AUC values are enclosed in parentheses.

plays an important role, we first study how the performance changes as a function of the network depth. Then, we study the performance variation for our model with respect to the batch size on both ABIDE and ADNI datasets.

Mitigating the Oversmoothing Problem. To evaluate the robustness of our approach to over-smoothing, we study the performance variation for our multi-layer AN-GCN model on the ABIDE and ADNI datasets with respect to the number of layers. Figure 4.8 shows how the node classification accuracy changes with the network's depth. As can be seen, the performance of AN-GCN

does not significantly degrade compared to GCN when the number of layers increases. Moreover, the performance gap between AN-GCN and GCN becomes quite noticeable when the network's depth rises. Hence, the classification performance of AN-GCN remains relatively stable as we increase the number of layers, demonstrating the robustness of our model against oversmoothing. This is largely due to the fact that the aggregation scheme of the proposed approach leverages residual connections to help alleviate the oversmoothing problem.

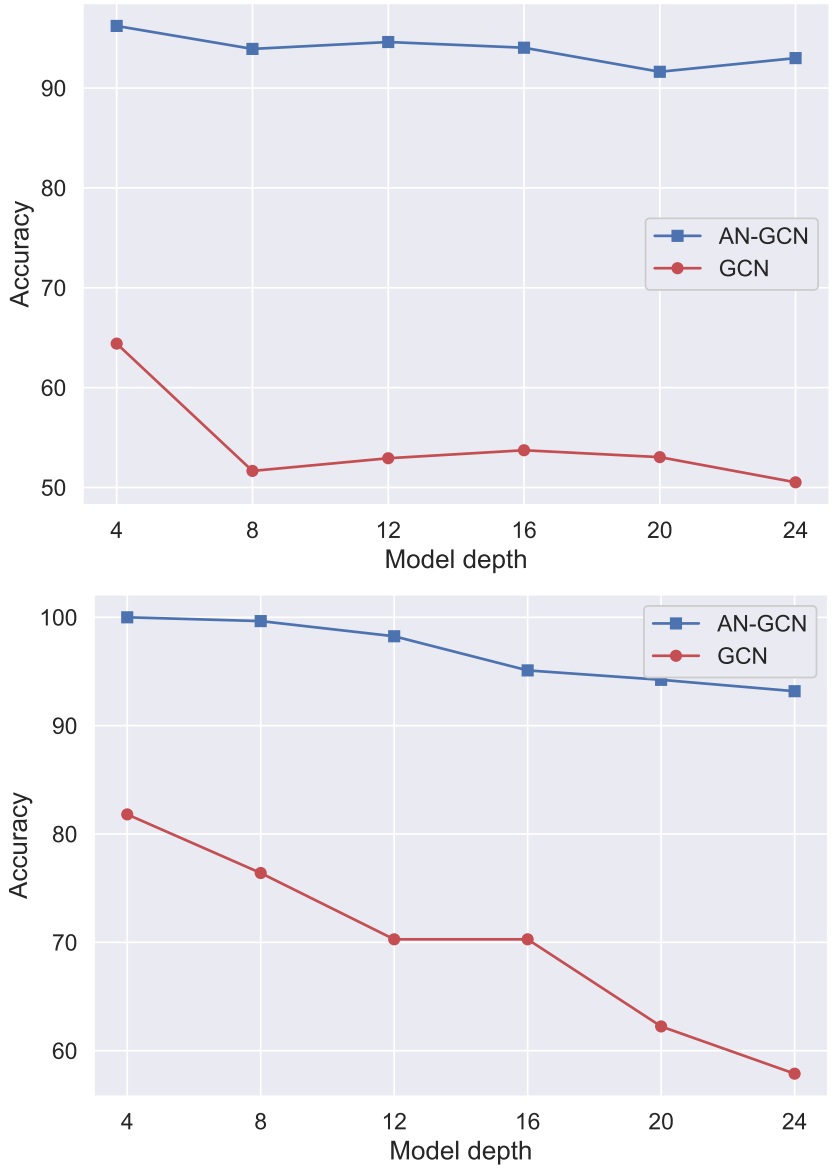

Figure 4.8: Performance comparison of AN-GCN and GCN on the ABIDE (top) and ADNI (bottom) datasets as we increase the number of layers.

Effect of Batch Size. We test the performance of our model using different values for the batch size on the ABIDE and ADNI datasets. As shown in Figure 4.9, the classification accuracy in-

creases rapidly at the beginning (i.e. for smaller batch sizes), reaching the highest value when the batch size is equal to 1000, and then slowly starts to decline on the ABIDE dataset or slows down on the ADNI dataset. This indicates that the batch size also plays an important role. In fact, we can observe that using a large batch size to train our model allows computational speedups from the parallelism of GPUs, but a larger batch size leads to poor generalization. It should also be pointed out that the drawback of using a smaller batch size is that the model is not guaranteed to converge to the global optimum.

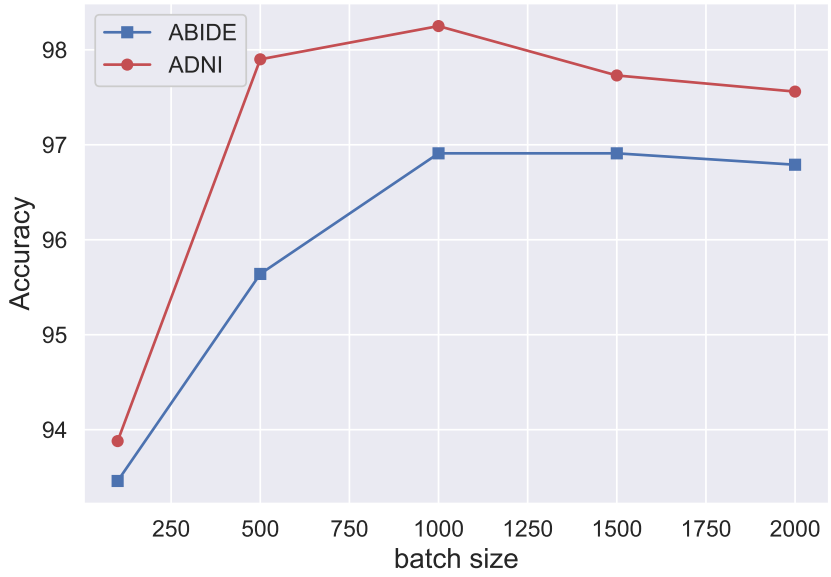

Figure 4.9: Sensitivity analysis of our model to the batch size on the ABIDE and ADNI datasets.

4.5 Conclusion

In this chapter, we introduced an end-to-end graph convolutional aggregation model by learning discriminative node representations from a population graph, consisting of subjects as nodes and edges as connections between subjects, with the goal to predict the status of each subject (i.e. diseased or healthy control) using imaging and non-imaging features associated to the graph nodes and edges, respectively. The proposed framework leverages skip connections and identity mapping, as well as aggregation in graph sampling in a bid to alleviate the problem of over-smoothing in graph convolutional networks. We demonstrated through extensive experiments that our model outperforms existing graph convolutional based methods for disease prediction on two large benchmark datasets, achieving significant relative improvements in classification accuracy over GCN and other strong baselines.

CHAPTER 5

Conclusions and Future Work

This thesis has presented several solutions to the carpal bone surface denoising, pediatric bone age assessment, and brain disorder prediction problems. We formulated our proposed surface denoising method as a constrained optimization whose objective function is comprised of a fidelity term specified by a noise model and a regularization term associated with data prior. We solved this constrained minimization problem iteratively using the conjugate gradient method, which is a commonly used iterative algorithm for solving sparse systems of linear equations. We also developed a unified deep learning framework for bone age assessment using instance segmentation and ridge regression in order to detect if bone maturity occurs at a rate compatible with the chronological (actual) age. We developed a regression network architecture for bone age assessment using a pre-trained deep learning model in conjunction with a regularized regression output layer. For radiograph segmentation, we leveraged Mask R-CNN, a region-based convolutional neural network for instance segmentation. In addition, we proposed an aggregator normalization graph convolutional network for developmental and brain disease prediction. We employed an aggregator normalization mechanism for feature propagation in an attempt to eliminate bias in minibatch estimation. We integrated both imaging and non-imaging features into the graph structure and formulated the disease prediction problem as a semi-supervised node classification on population graphs. Experimental results demonstrated that the proposed approaches outperform existing baseline methods in the literature. In Section 5.1, we summarize the contributions made in each of the previous chapters, as well as the conclusions obtained from the associated research work. In Section 5.2, we discuss the limitations of the proposed approaches. Finally, we point out future research directions related to this thesis in Section 5.3.

5.1 Contributions of the Thesis

5.1.1 Carpal Bone Surface Denoising

In Chapter 2, we developed a feature-preserving approach to carpal bone surface denoising. The proposed method is formulated as a constrained optimization problem whose regularized objective function includes a fidelity term determined by a noise model and a regularization term specified by the prior data. Minimizing the fidelity terms yields a solution as close as possible to the input, while minimizing the regularization term yields a smooth solution. Both terms are weighted by a normalized mesh Laplacian, which is defined in terms of a data-adaptive kernel similarity matrix together with matrix balancing. The qualitative and quantitative evaluation results demonstrated that our approach offers superior performance over existing mesh denoising techniques.

5.1.2 Pediatric Bone Age Assessment

In Chapter 3, we introduced a unified framework for pediatric bone age assessment by leveraging instance segmentation and ridge regression. We presented a hand segmentation and annotation model for radiographic images, followed by background removal. In order to learn salient features from the segmented radiographs, we designed a regression neural network architecture composed of a pre-trained convolutional neural network. Extensive experiments demonstrated the competitive performance of the proposed framework in comparison with baseline approaches for bone age assessment.

5.1.3 Classification of Developmental and Brain Disorders

In Chapter 4, we proposed an aggregator normalization graph convolutional network with skip connections and identity mapping with the goal to detect neurodevelopmental and neurodegenerative brain disorders, as well as to predict the status of each subject (i.e. diseased or healthy control). We formulated the disease prediction problem as a semi-supervised node classification on population graphs. The proposed approach learns discriminative graph node representations by incorporating both imaging and non-imaging features into the nodes and edges of the graph, respectively. Each graph node represents a subject with an associated feature vector extracted from imaging data, and each edge captures similarities between a pair of subjects with non-imaging data integrated into the edge weight. We demonstrated through extensive experiments that our proposed model outperforms strong baselines on two large benchmark datasets for the prediction of autism spectrum disorder and Alzheimer's disease.

5.2 Limitations

While the proposed denoising method is able to remove undesirable noise while preserving important geometric features of carpal bone surfaces in an effective manner, it does, however, rely on a matrix balancing procedure, which requires matrix inversion to find the filtering matrix. The limitation may reduce the computational efficiency of our surface denoising algorithm, particularly on large meshes with thousands of nodes.

For bone age assessment, the main limitation of using Mask R-CNN for instance segmentation is that it treats part of the background in an image as foreground, resulting in inaccurate target segmentation. Also, Mask R-CNN often fails to detect objects when motion blur is present. On the other hand, the proposed aggregator normalization graph convolutional network for disease prediction employs first-order graph convolutions; thereby, it does not capture long-range dependencies between subjects in a population graph. Using multi-hop neighborhoods for node feature aggregation can help not only in capturing the long-range dependencies between subjects, but also in alleviating the oversmoothing problem, where repeated graph convolutions make learned node embeddings indistinguishable.

5.3 Future Work

Several interesting research directions, motivated by this thesis, are discussed below:

5.3.1 Graph Convolutional Networks for Carpal Bone Surface Denoising

We plan to explore the use of graph convolutional networks (GCNs) for carpal bone surface denoising. The main idea is to learn a graph representation for each surface patch, followed by graph convolution, face normal refinement of the surface patches, and a vertex updating scheme defined in terms of neighboring faces and denoised face normals. We also intend to incorporate edge-aware filters to tackle data-driven geometry processing problems.

5.3.2 Graph Convolutional Networks for Brain Age Prediction

Understanding how the brain develops is vital to designing prediction models for a multitude of developmental disorders and degenerative neurological disorders such as autism spectrum disorder, Alzheimer's disease, dementia and multiple sclerosis. The gap between the estimated brain age and the chronological age of a patient can help predict the risk of these disorders with good accuracy in seemingly healthy patients. To address this challenging problem of brain age estimation, we

plan to use populations graphs in conjunction with a higher-order aggregator normalization graph convolutional network (AN-GCN) by integrating both imaging and nonimaging features into the graph nodes and edges, respectively. We also intend to explore the robustness of the AN-GCN architecture to node noise and edge sparsity in a bid to estimate the generalizability of graph convolutional networks to real clinical settings. The key idea is to perform brain age estimation by fitting a model that predicts the chronological age for healthy subjects in the training set, and then applying it to the remaining population in the test set.

5.3.3 Spatial-Temporal Graph Convolutional Networks for Gait Recognition

Abnormal gaits are typically caused by geriatric degenerative and neurological disorders, such as Alzheimer's disease, Parkinson's disease and stroke, and can result in a lower quality of life for patients [105]. The human body skeleton can be modeled as a graph, where body joints represent nodes and bones represent edges. We aim to apply spatial-temporal graph convolutional networks to skeleton-based abnormal gait recognition. More precisely, we intend to model the structural information between body joints along both the spatial and temporal dimensions by defining a gait sequence as an undirected graph. To this end, we plan to tackle the gait recognition problem from a short sequence of 2D joint detections by exploiting spatial and temporal relationships. The edge set of the graph consists of spatial and temporal connections. The former include both direct and indirect kinematic dependencies in each frame, while the latter link each joint to its counterpart in the neighboring frames.

References

[1] A. Kheradmand and P. Milanfar, "A general framework for regularized, similarity-based image restoration," *IEEE Transactions on Image Processing*, pp. 5136–5151, 2014.

[2] W. Greulich and S. Pyle, *Radiographic Atlas of Skeletal Development of the Hand and Wrist*. Stanford University Press, 1959.

[3] J. Tanner, R. Whitehouse, N. Cameron, and W. Marshall, *Assessment of skeletal maturity and prediction of adult height (TW2 method)*. Academic Press London, 1975.

[4] S. Parisot, S. I. Ktena, E. Ferrante, M. Lee, R. Guerrero, B. Glocker, and D. Rueckert, "Disease prediction using graph convolutional networks: application to autism spectrum disorder and Alzheimer's disease," *Medical Image Analysis*, pp. 117–130, 2018.

[5] I. Salim and A. Ben Hamza, "Fast feature-preserving approach to carpal bone surface denoising," *Sensors*, 2018.

[6] I. Salim and A. Ben Hamza, "Ridge regression neural network for pediatric bone age assessment," *Multimedia Tools and Applications*, pp. 30461–30478, 2021.

[7] G. Taubin, "A signal processing approach to fair surface design," in *Proc. ACM SIGGRAPH*, pp. 351–358, 1995.

[8] J. Wang, X. Zhang, and Z. Yu, "A cascaded approach for feature-preserving surface mesh denoising," *Computer-Aided Design*, pp. 597–610, 2012.

[9] L. Zhu, M. Wei, J. Yu, W. Wang, J. Qin, and P.-A. Heng, "Coarse-to-fine normal filtering for feature-preserving mesh denoising based on isotropic subneighborhoods," in *Proc. Computer Graphics Forum*, pp. 371–380, 2013.

[10] S. Fleishman, I. Drori, and D. Cohen-Or, "Bilateral mesh denoising," in *Proc. ACM SIGGRAPH*, pp. 950–953, 2003.

[11] Y. Zheng, H. Fu, O. K.-C. Au, and C.-L. Tai, "Bilateral normal filtering for mesh denoising," *IEEE Trans. Visualization and Computer Graphics*, pp. 1521–1530, 2011.

[12] W. Zhang, B. Deng, J. Zhang, S. Bouaziz, and L. Liu, "Guided mesh normal filtering," *Computer Graphics Forum*, pp. 23–34, 2015.

[13] V. Gilsanz, , and O. Ratib, *Hand Bone Age: A Digital Atlas of Skeletal Maturity*. Springer, 2012.

[14] H. Thodberg, S. Kreiborg, A. Juul, and K. Pedersen, "The BoneXpert method for automated determination of skeletal maturity," *IEEE Transactions on Medical Imaging*, vol. 28, no. 1, pp. 52–66, 2009.

[15] T. Van Steenkiste, J. Ruyssinck, O. Janssens, B. Vandersmissen, F. Vandecasteele, P. Devolder, E. Achten, S. Van Hoecke, D. Deschrijver, and T. Dhaene, "Automated assessment of bone age using deep learning and Gaussian process regression," in *Proc. Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pp. 674–677, 2018.

[16] V. Iglovikov, A. Rakhlin, A. Kalinin, and A. Shvets, "Pediatric bone age assessment using deep convolutional neural networks," in *Proc. Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*, pp. 300–308, 2018.

[17] E. Wu, B. Kong, X. Wang, J. Bai, Y. Lu, F. Gao, S. Zhang, K. Cao, Q. Song, S. Lyu, and Y. Yin, "Residual attention based network for hand bone age assessment," in *Proc. IEEE International Symposium on Biomedical Imaging*, pp. 1158–1161, 2019.

[18] S. J. Son, Y. Song, N. Kim, Y. Do, N. Kwak, M. S. Lee, and B.-D. Lee, "TW3-based fully automated bone age assessment system using deep neural networks," *IEEE Access*, pp. 33346–33358, 2019.

[19] M. Xu, V. Calhoun, R. Jiang, W. Yan, and J. Sui, "Brain imaging-based machine learning in autism spectrum disorder: methods and applications," *Journal of Neuroscience Methods*, p. 109271, 2021.

[20] B. Lei, E. Liang, M. Yang, P. Yang, F. Zhou, E.-L. Tan, Y. Lei, C.-M. Liu, T. Wang, X. Xiao, *et al.*, "Predicting clinical scores for alzheimer's disease based on joint and deep learning," *Expert Systems with Applications*, p. 115966, 2022.

[21] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. International Conference on Learning Representations*, 2017.

[22] B. Xu, H. Shen, Q. Cao, Y. Qiu, and X. Cheng, "Graph wavelet neural network," in *Proc. International Conference on Learning Representations*, 2019.

[23] H. Zeng, H. Zhou, A. Srivastava, R. Kannan, and V. Prasanna, "GraphSAINT: Graph sampling based inductive learning method," in *Proc. International Conference on Learning Representations*, 2020.

[24] M. Chen, Z. Wei, Z. Huang, B. Ding, and Y. Li, "Simple and deep graph convolutional networks," in *Proc. International Conference on Machine Learning*, pp. 1725–1735, 2020.

[25] A. Kazi, S. shekarforoush, S. krishna, H. Burwinkel, G. Vivar, K. Kortuem, S.-A. Ahmadi, S. Albarqouni, and N. Navab, "InceptionGCN: Receptive field aware graph convolutional network for disease prediction," in *Proc. International Conference on Information Processing in Medical Imaging*, 2019.

[26] L. Cosmo, A. Kazi, S.-A. Ahmadi, N. Navab, and M. Bronstein, "Latent-graph learning for disease prediction," in *Proc. International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 643–653, 2020.

[27] M. Cao, M. Yang, C. Qin, X. Zhu, Y. Chen, J. Wang, and T. Liu, "Using deepGCN to identify the autism spectrum disorder from multi-site resting-state data," *Biomedical Signal Processing and Control*, p. 103015, 2021.

[28] S. Zheng, Z. Zhu, Z. Liu, Z. Guo, Y. Liu, and Y. Zhao, "Multi-modal graph learning for disease prediction," *IEEE Transactions on Medical Imaging*, 2022.

[29] L. Pan, J. Liu, M. Shi, C. W. Wong, and K. H. K. Chan, "Identifying autism spectrum disorder based on individual-aware down-sampling and multi-modal learning," *arXiv preprint arXiv:2109.09129*, 2021.

[30] D. Yao, J. Sui, M. Wang, E. Yang, Y. Jiaerken, N. Luo, P. T. Yap, M. Liu, and D. Shen, "A mutual multi-scale triplet graph convolutional network for classification of brain disorders using functional or structural connectivity," *IEEE Transactions on Medical Imaging*, pp. 1279–1289, 2021.

[31] G. Wen, P. Cao, H. Bao, W. Yang, T. Zheng, and O. Zaiane, "MVS-GCN: A prior brain structure learning-guided multi-view graph convolution network for autism spectrum disorder diagnosis," *Computers in Biology and Medicine*, 2022.

[32] Y. LeCun, B. Boser, J. Denker, D. Henderson, R. Howard, W. Hubbard, and L. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Computation*, vol. 1, no. 4, pp. 541–551, 1989.

[33] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. International Conference on Learning Representations*, 2015.

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

[35] J. S. Suri, D. L. Wilson, and S. Laxminarayan, *Handbook of biomedical image analysis*, vol. 2. 2005.

[36] A. Fathi, Z. Wojna, V. Rathod, P. Wang, H. O. Song, S. Guadarrama, and K. P. Murphy, "Semantic instance segmentation via deep metric learning," *arXiv preprint arXiv:1703.10277*, 2017.

[37] B.-U. Bae, W. Bae, and K.-H. Jung, "Improved deep learning model for bone age assessment using triplet ranking loss," in *1st Conference on Medical Imaging with Deep Learning*, 2018.

[38] O. Ronneberger, P. Fischer, and T. Brox, "U-net: convolutional networks for biomedical image segmentation," in *Proc. International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 234–241, 2015.

[39] T. Falk, D. Mai, R. Bensch, Ö. Çiçek, A. Abdulkadir, Y. Marrakchi, A. Böhm, J. Deubner, Z. Jäckel, K. Seiwald, *et al.*, "U-net: deep learning for cell counting, detection, and morphometry," *Nature Methods*, vol. 16, no. 1, p. 67, 2019.

[40] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proc. IEEE International Conference on Computer Vision*, pp. 2961–2969, 2017.

[41] A. Ortega, P. Frossard, J. Kovačević, J. M. Moura, and P. Vandergheynst, "Graph signal processing: Overview, challenges, and applications," *Proceedings of the IEEE*, pp. 808–828, 2018.

[42] C. Luo, F. Li, P. Li, C. Yi, C. Li, Q. Tao, X. Zhang, Y. Si, D. Yao, G. Yin, *et al.*, "A survey of brain network analysis by electroencephalographic signals," *Cognitive Neurodynamics*, pp. 17–41, 2022.

[43] H. Onias, A. Viol, F. Palhano-Fontes, K. C. Andrade, M. Sturzbecher, G. Viswanathan, and D. B. de Araujo, "Brain complex network analysis by means of resting state fmri and graph analysis: Will it be helpful in clinical epilepsy?," *Epilepsy & Behavior*, pp. 71–80, 2014.

[44] M. Botsch, L. Kobbelt, M. Pauly, P. Alliez, and B. Levy, *Polygon Mesh Processing*. AK Peters/CRC Press, 2010.

[45] H. Yagou, Y. Ohtake, and A. Belyaev, "Mesh smoothing via mean and median filtering applied to face normals," in *Proc. Geometric Modeling and Processing*, pp. 124–131, 2002.

[46] T. Tasdizen, R. Whitaker, P. Burchard, and S. Osher, "Geometric surface smoothing via anisotropic diffusion of normals," in *Proc. IEEE Visualization*, pp. 125–132, 2002.

[47] U. Clarenz, U. Diewald, , and M. Rumpf, "Processing textured surfaces via anisotropic geometric diffusion," *IEEE Transactions on Image Processing*, vol. 13, no. 2, pp. 248–261, 2004.

[48] Y. Zhang and A. Ben Hamza, "Vertex-based diffusion for 3d mesh denoising," *IEEE Transactions on Image Processing*, vol. 16, no. 4, pp. 1036–1045, 2007.

[49] J. Weickert, *Anisotropic diffusion in image processing*. Teubner-Verlag, 1998.

[50] D. Barash, "A fundamental relationship between bilateral filtering, adaptive smoothing, and the nonlinear diffusion equation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 844–847, 2002.

[51] S. Paris, P. Kornprobst, J. Tumblin, and F. Durand, "A gentle introduction to bilateral filtering and its applications," *ACM SIGGRAPH courses*, pp. 1:1–1:50, 2007.

[52] X. Sun, P. L. Rosin, R. R. Martin, and F. C. Langbein, "Fast and effective feature-preserving mesh denoising," *IEEE Transactions on Visualization and Computer Graphics*, pp. 925–938, 2007.

[53] H. Huang and U. Ascher, "Fast denoising of surface meshes with intrinsic texture," *Inverse Problems*, 2008.

[54] A. F. El Ouafdi, D. Ziou, and H. Krim, "A smart stochastic approach for manifolds smoothing," *Computer Graphics Forum*, pp. 1357–1364, 2008.

[55] S. K. Yadav, U. Reitebuch, and K. Polthier, "Robust and high fidelity mesh denoising," *IEEE Transactions on Visualization and Computer Graphics*, 2018.

[56] A. Elmoatz, O. Lezoray, and S. Bougleux, "Nonlocal discrete regularization on weighted graphs: A framework for image and manifold processing," *IEEE Transactions on Image Processing*, pp. 1047–1060, 2008.

[57] M. K. Chung, Y. Wang, and G. Wu, "Heat kernel smoothing in irregular image domains," in *International Conference of the IEEE Engineering in Medicine and Biology Society*, 2018.

[58] M. Chung, A. Qiu, S. Seo, and H. Vorperian, "Unified heat kernel regression for diffusion, kernel smoothing and wavelets on manifolds and its application to mandible growth modeling in CT images," *Medical Image Analysis*, pp. 63–76, 2015.

[59] R. Sinkhorn and P. Knopp, "Concerning nonnegative matrices and doubly stochastic matrices," *Pacific Journal of Mathematics*, pp. 343–348, 1967.

[60] P. A. Knight and D. Ruiz, "A fast algorithm for matrix balancing," *IMA Journal of Numerical Analysis*, pp. 1029–1047, 2013.

[61] M. Masoumi, M. Rezaei, and A. Ben Hamza, "Global spectral graph wavelet signature for surface analysis of carpal bones," *Physics in Medicine & Biology*, 2018.

[62] D. Martin, J. Wit, Z. Hochberg, L. Savendahl, R. van Rijn, O. Fricke, N. Cameron, J. Caliebe, T. Hertel, D. Kiepe, K. Albertsson-Wikland, H. Thodberg, G. Binder, and M. Ranke, "The use of bone age in clinical practice - part 1," *Hormone Research in Paediatrics*, vol. 76, no. 1, pp. 1–9, 2011.

[63] M. Satoh, "Bone age: assessment methods and clinical applications," *Clinical Pediatric Endocrinology*, vol. 24, no. 4, pp. 143–152, 2015.

[64] K. Somkantha, N. Theera-Umpon, and S. Auephanwiriyakul, "Bone age assessment in young children using automatic carpal bone feature extraction and support vector regression," *Journal of Digital Imaging*, vol. 24, no. 6, pp. 1044–1058, 2011.

[65] K. Alshamrani and A. Offiah, "Applicability of two commonly used bone age assessment methods to twenty-first century UK children," *European Radiology*, pp. 1–10, 2019.

[66] Y. Bengio, "Learning deep architectures for AI," *Foundations and Trends in Machine Learning*, vol. 2, no. 1, pp. 1–127, 2009.

[67] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks*, vol. 61, pp. 85–117, 2015.

[68] S. Halabi, L. Prevedello, J. Kalpathy-Cramer, A. Mamonov, A. Bilbily, M. Cicero, I. Pan, L. Pereira, R. Sousa, N. Abdala, F. Kitamura, H. Thodberg, L. Chen, G. Shih, K. Andriole, M. Kohli, B. Erickson, and A. A.E. Flanders, "The RSNA pediatric bone age machine learning challenge," *Radiology*, vol. 290, no. 2, pp. 498–503, 2019.

[69] C. Spampinato, S. Palazzo, D. Giordano, M. Aldinucci, and R. Leonardi, "Deep learning for automated skeletal bone age assessment in X-ray images," *Medical Image Analysis*, vol. 36, pp. 41–51, 2017.

[70] H. Lee, S. Tajmir, J. Lee, M. Zissen, B. A. Yeshiwas, T. K. Alkasab, G. Choy, and S. Do, "Fully automated deep learning system for bone age assessment," *Journal of Digital Imaging*, vol. 30, no. 4, pp. 427–441, 2017.

[71] D. B. Larson, M. C. Chen, M. P. Lungren, S. S. Halabi, N. V. Stence, and C. P. Langlotz, "Performance of a deep-learning neural network model in assessing skeletal maturity on pediatric hand radiographs," *Radiology*, pp. 313–322, 2017.

[72] C. Tong, B. Liang, J. Li, and Z. Zheng, "A deep automated skeletal bone age assessment model with heterogeneous features learning," *Journal of Medical Systems*, vol. 42, no. 12, p. 249, 2018.

[73] X. Chen, J. Li, Y. Zhang, Y. Lu, and S. Liu, "Automatic feature extraction in X-ray image based on deep learning approach for determination of bone age," *Future Generation Computer Systems*, pp. 1–7, 2019.

[74] B. Liu, Y. Zhang, M. Chu, X. Bai, and F. Zhou, "Bone age assessment based on rank-monotonicity enhanced ranking CNN," *IEEE Access*, vol. 7, pp. 120976–120983, 2019.

[75] X. Pan, Y. Zhao, H. Chen, D. Wei, C. Zhao, and Z. Wei, "Fully automated bone age assessment on large-scale hand X-ray dataset," *International Journal of Biomedical Imaging*, 2020.

[76] R. Liu, Y. Jia, X. He, Z. Li, J. Cai, H. Li, , and X. Yang, "Pediatric hand radiograph segmentation for bone age assessment," *International Journal of Biomedical Imaging*, 2020.

[77] A. Wibisono and P. Mursanto, "Multi region-based feature connected layer (RB-FCL) of deep learning models for bone age assessment," *Journal of Big Data*, 2020.

[78] D. Omeiza, S. Speakman, C. Cintas, and K. Weldermariam, "Smooth Grad-CAM++: An enhanced inference level visualization technique for deep convolutional neural network models," *arXiv:1908.01224*, 2019.

[79] R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: visual explanations from deep networks via gradient-based localization," in *Proc. IEEE International Conference on Computer Vision*, pp. 618–626, 2017.

[80] D. Smilkov, N. Thorat, B. Kim, F. Viégas, and M. Wattenberg, "SmoothGrad: removing noise by adding noise," in *Proc. ICML workshop on visualization for deep*, 2017.

[81] A. Chattopadhay, A. Sarkar, P. Howlader, and V. Balasubramanian, "Grad-CAM++: Generalized gradient-based visual explanations for deep convolutional networks," in *Proc. IEEE Winter Conference on Applications of Computer Vision*, 2018.

[82] Y.-B. Wang, Z.-H. You, X. Li, T.-H. Jiang, X. Chen, X. Zhou, and L. Wang, "Predicting protein–protein interactions from protein sequences by a stacked sparse autoencoder deep neural network," *Molecular BioSystems*, pp. 1336–1344, 2017.

[83] T. R. Insel and B. N. Cuthbert, "Brain disorders? precisely," *Science*, pp. 499–500, 2015.

[84] F. Wu, A. Souza, T. Zhang, C. Fifty, T. Yu, and K. Weinberger, "Simplifying graph convolutional networks," in *Proc. International Conference on Machine Learning*, pp. 6861–6871, 2019.

[85] M. Khosla, K. Jamison, G. H. Ngo, A. Kuceyeski, and M. R. Sabuncu, "Machine learning in resting-state fmri analysis," *Magnetic Resonance Imaging*, pp. 101–121, 2019.

[86] K. Gopinath, C. Desrosiers, and H. Lombaert, "Graph convolutions on spectral embeddings for cortical surface parcellation," *Medical Image Analysis*, pp. 297–305, 2019.

[87] C. Su, J. Tong, Y. Zhu, P. Cui, and F. Wang, "Network embedding in biomedical data science," *Briefings in Bioinformatics*, pp. 182–197, 2020.

[88] X. Yue, Z. Wang, J. Huang, S. Parthasarathy, S. Moosavinasab, Y. Huang, S. M. Lin, W. Zhang, P. Zhang, and H. Sun, "Graph embedding on biomedical networks: methods, applications and evaluations," *Bioinformatics*, pp. 1241–1251, 2020.

[89] J. Yang, Q. Zhu, R. Zhang, J. Huang, and D. Zhang, "Unified brain network with functional and structural data," in *Proc. International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 114–123, 2020.

[90] L. Goldsberry, W. Huang, N. F. Wymbs, S. T. Grafton, D. S. Bassett, and A. Ribeiro, "Brain signal analytics from graph signal processing perspective," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 851–855, 2017.

[91] S. I. Ktena, S. Parisot, E. Ferrante, M. Rajchl, M. Lee, B. Glocker, and D. Rueckert, "Metric learning with spectral graph convolutions on brain connectivity networks," *NeuroImage*, pp. 431–442, 2018.

[92] G. Ma, N. K. Ahmed, T. L. Willke, D. Sengupta, M. W. Cole, N. B. Turk-Browne, and P. S. Yu, "Deep graph similarity learning for brain data analysis," in *Proc. ACM International Conference on Information and Knowledge Management*, pp. 2743–2751, 2019.

[93] Q. Li, Z. Han, and X. Wu, "Deeper insights into graph convolutional networks for semi-supervised learning," in *Proc. AAAI Conference on Artificial Intelligence*, pp. 3538–3545, 2018.

[94] K. Xu, C. Li, Y. Tian, T. Sonobe, K. ichi Kawarabayashi, and S. Jegelka, "Representation learning on graphs with jumping knowledge networks," in *Proc. International Conference on Machine Learning*, 2018.

[95] L. Zhao and L. Akoglu, "PairNorm: Tackling oversmoothing in GNNs," in *Proc. International Conference on Learning Representations*, 2020.

[96] Y. Rong, W. Huang, T. Xu, and J. Huang, "DropEdge: Towards deep graph convolutional networks on node classification," in *Proc. International Conference on Learning Representations*, 2020.

[97] H. Jiang, P. Cao, M. Xu, J. Yang, and O. Zaiane, "HI-GCN: A hierarchical graph convolution network for graph embedding learning of brain network and brain disorders prediction," *Computers in Biology and Medicine*, pp. 1–16, 2020.

[98] Y. Huang and A. C. S. Chung, "Diffusion improves graph learning," in *Proc. International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 13354–13366, 2019.

[99] J. C. Paetzold, J. McGinnis, S. Shit, I. Ezhov, P. Büschl, C. Prabhakar, M. I. Todorov, A. Sekuboyina, G. Kaissis, A. Ertürk, *et al.*, "Whole brain vessel graphs: A dataset and benchmark for graph learning and neuroscience (vesselgraph)," in *35th Conference on Neural Information Processing Systems (NeurIPS 2021) Track on Datasets and Benchmarks*, 2021.

[100] Y. Li and Y. Yuan, "Convergence analysis of two-layer neural networks with relu activation," in *Advances in Neural Information Processing Systems*, pp. 597–607, 2017.

[101] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. International Conference on Learning Representations*, 2015.

[102] Y. Huang and A. C. S. Chung, "Edge-variational graph convolutional networks for uncertaintyaware disease prediction," in *Proc. International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 562–572, 2020.

[103] P. Gu, X. Xu, Y. Luo, P. Wang, and J. Lu, "BCN-GCN: A novel brain connectivity network classification method via graph convolution neural network for Alzheimer's disease," in *Proc. International Conference on Neural Information Processing*, pp. 657–668, 2021.

[104] W. Yu, B. Lei, M. K. Ng, A. C. Cheung, Y. Shen, and S. Wang, "Tensorizing GAN with high-order pooling for Alzheimer's disease assessment," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–15, 2021.

[105] H. Tian, X. Ma, H. Wu, and Y. Li, "Skeleton-based abnormal gait recognition with spatio-temporal attention enhanced gait-structural graph convolutional networks," *Neurocomputing*, pp. 116–126, 2022.