Deep Learning for Multi-Channel Image Analysis with Applications to Remote Sensing and Biomedicine

by
Farideh Foroozandeh Shahraki

A Dissertation submitted to the Department of Electrical and Computer Engineering,
Cullen College of Engineering
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

in Electrical Engineering

Chair of the Committee: Dr. Saurabh Prasad

Committee Member: Dr. Badrinath Roysam

Committee Member: Dr. David Mayerich

Committee Member: Dr. Rohith Reddy

Committee Member: Dr. Dragan Maric

University of Houston

May 2022

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my advisor, Dr. Saurabh Prasad , who provided technical and moral support during my doctoral study. I would further like to thank the rest of the committee members, Dr. Badrinath Roysam Dr. David Mayerich, Dr. Rohith Reddy and Dr. Dragan Maric for their guidance and insightful feedback.

I would like to thank my parents, Mrs. Fereshteh Nekooei Shahraki and Mr. Daryoosh Foroozandeh Shahraki, for their support, patience and encouraging me to pursue my dream, my Husband, Ebrahim, for supporting me during my doctoral study. I could have not made it without you. Finally, special thanks goes to my friends and collaborators at ECE department. I enjoyed every moment of working with you and learned a lot from all of you.

Farideh Foroozandeh Shahraki

ABSTRACT

Deep neural networks are emerging as a popular choice for multi-channel image analysis – compared with other machine learning approaches, they have been shown to be more effective for a variety of applications in hyperspectral (HSI) and multispectral imaging. We focus on application specific nuances and design choices with respect to deploying such networks for robust analysis of hyperspectral and multispectral images. We provide quantitative and qualitative results with a variety of deep learning architectures in remote sensing, biomedical FTIR and multiplex rat brain images. In this work, not only are traditional deep learning models such as Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and Convolutional-Recurrent Neural Networks (CRNNs) investigated, we also design and develop Graph based convolutional neural networks (GCNs) with applications to hyperspectral images. To optimize GCNs for HSI data, we proposed a new method of adjacency matrix construction (a semi-supervised adjacency matrix) which leverages class specific and cluster specific properties of the underlying imagery data. Finally, we designed a semi-automatic pipeline that utilizes registration and deep semantic segmentation for aligning (fitting) a brain atlas on multiplex rat brain images.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

Hyperspectral imaging (HSI) combines spectroscopic instrumentation with imaging systems to provide spatially-resolved spectroscopic data. HSI instrumentation can acquire hundreds or thousands of spectra in a $X \times Y \times Z$ data cube, where X and Y are spatial dimensions and Z describes spectral content (Fig. 1.1). Information encoded along the spectral dimension depends on modality, with the most common approaches being ultraviolet [6], visible [7], near-infrared [8], and vibrational [9] spectroscopy. Non-optical methods include mass [10] and nuclear magnetic resonance (NMR) spectroscopy [11]. The encoded spectral signature provides insight into the material composition at each $[x, y]^T$ spatial location, where $x \in X$ and $y \in Y$. This spectral signature provides a fingerprint for material identification and quantifiable properties such as density, absorbance, and emission. HSI approaches have seen broad use in remote sensing [8], biomedicine [9], astronomy [12], agriculture and food quality [13, 14], and pharmaceuticals [15].

Deep neural networks have emerged as a robust machine learning set of tools for computer vision tasks related to analysis of color images. The suitability of convolution neural networks, recurrent neural networks and their variants for analysis of color imagery and video is well documented and understood in the numerous developments in the field. However, the fields of remote sensing and biomedical image processing often rely on more than three channel optical imagery for the underlying analysis tasks – specifically, hyperspectral imagery is commonly used for robust material-specific characterization of each pixel in the image. Deep learning can facilitate robust image analysis of such multi-channel optical data, but the network architectures and design

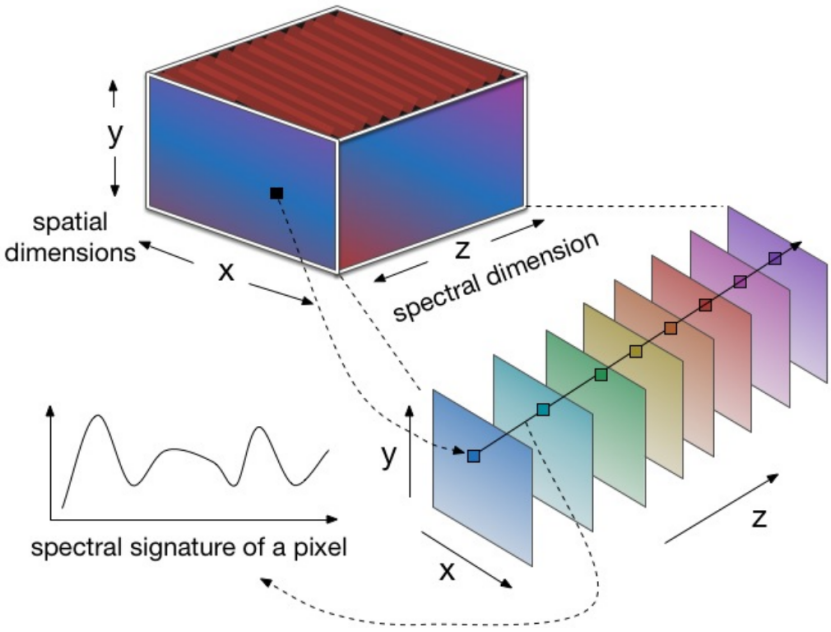


Figure 1.1: Structure of an HSI data cube. The measured data in HSI can be visualized as a data cube. Each slice of the data cube contains an image of the scene at a particular wavelength. Each pixel is associated with a vector of spectral responses otherwise known as a spectral signature.

choices often would need to tailored to the unique characteristics of such data. This dissertation proposes image analysis for multi-channel imagery.

In this work, various deep learning classification and segmentation methods have been designed and studied for Hyperspectral remote sensing imagery, FTIR microscopy imagery, and multiplex brain-tissue imagery. Chapter 2 summarizes our study in practical applications of deep learning in both remote sensing and FTIR biomedical HSI. In this chapter, we focus on data collection and preprocessing, parameter tuning, and practical considerations for selecting appropriate deep learning architectures, and this work published in [16] and [17]. In chapter 3, we design graph-based CNN architectures for classification of hyperspectral data in remote sensing and biomedical hyperspectral images. In this work, we propose a semi-supervised strategy to construct a robust adjacency matrix that is used by the graph convolutional networks which has been published in [1]. We also developed a spatial-spectral fusion

network where spatial convolutions are utilized in conjunction with graph convolutions along the spectral reflectance direction to learn both object-specific spatial features and the graph-structure of the spectral features simultaneously which has been published in [18]. In chapter 4, we combine image registration and segmentation methodologies to build a semi-automatic atlas fitting for multiplex rat brain images. In this chapter, we design various semantic segmentation models to leverage this characteristic of brain which different types of cells exhibit different image characteristics in different brain regions.

Chapter 2

Convolutional Neural Networks for Hyperspectral Image Analysis

Convolutional neural networks (CNNs) are specialized feed-forward neural networks for processing data sampled on a uniform grid, such as an image. In the case of HSI, this can include 1D spectral sampling, 2D spatial sampling, or 3D sampling of the entire image tensor. Each CNN layer generates a higher-level abstraction of the input data, generally called a "featuremap", that preserves essential and unique information. CNNs are able to achieve superior performance by employing a deep hierarchy of layers, and have recently become a popular deep learning method achieving significant success in hyperspectral pixel classification [19, 20, 21], scene understanding [22], target detection [23, 24], and anomaly detection [25].

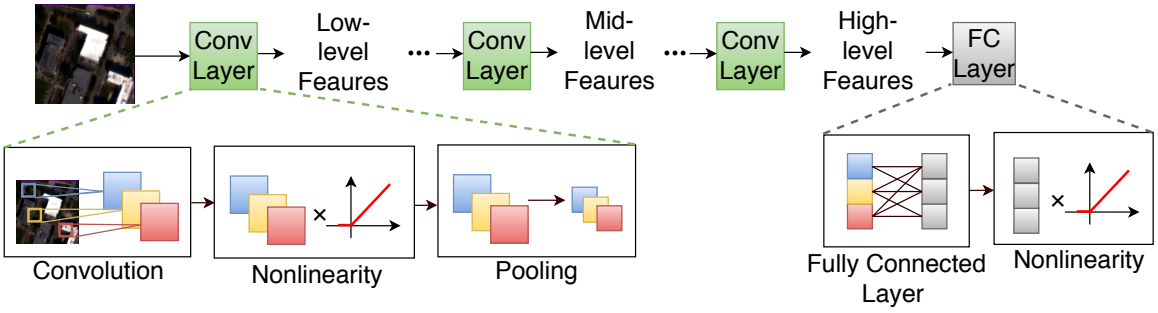


Figure 2.1: An architecture of a Convolutional Neural Network (CNN). Each convolution layer consists of convolution, nonlinearity, and pooling operations, which generates higher level feature of the input. The first few layers generate low-level features, middle layers generate mid-level features, and the last few layers generate high-level features which are fed into fully connected layer.

2.1 Building Blocks of CNNs

CNNs are trainable multilayer architectures composed of multiple feature-extraction stages. Each stage consists of three layers: 1) a convolutional, 2) nonlinearity, and 3) pooling. A typical CNN is composed of some feature-extraction stages followed by one or more fully connected layers and final classifier layer as shown in Fig. 2.1. Each part of a typical CNN is described in the following sections.

Convolutions A CNN layer performs several convolutions in parallel to produce a set of linear activations. Convolutional layers are responsible for extracting local features at different positions using trainable kernels ${}^{(l)}_{i,j}$ that act as connection weights between feature map i of layer $l-1$ and featuremap j of layer l. The units at convolution layer l compute activations ${}^{l}_{j}$ based on spatially contiguous units in the feature map ${}^{l-1}_{i}$ of layer $l-1$ by convolving the kernels ${}^{l}_{i,j}$

$$ {}^{(l)}_{j} = f\left(\sum_{i=1}^{M^{(l-1)}} {}^{(l-1)}_{i} *^{(l)}_{i,j} + b^{(l)}_{j}\right) \tag{2.1} $$

where $M^{(l-1)}$ denotes the number of feature maps in the layer of $l-1$, $*$ is convolution operator, $b^{(l)}_{j}$ is a bias parameter, and $f(\cdot)$ represents a nonlinear activation function.

Pooling A *pooling function* reduces the dimensionality of a feature map and is applied to each data channel (or band) to reduce sensitivity to rotation, translation, and scaling. Pooling functions also aggregate responses within and across feature maps. The pooling function combines a set of values within a receptive field (that defines a spatially local neighborhood, as set by the filter-size) into fewer values and can be configured based on the size of the receptive field (e.g., 2×2) and selected pooling operation (e.g., max or average). The max pooling function applies a window function to the input patch and computes the maximum within the neighborhood to preserve texture information. The average pooling function calculates the mean

of the input elements within a patch to preserve background information. Pooling is typically performed on non-overlapping blocks, however some methods however this is not required [26]. In general, non-overlapping pooling is used for dimension reduction of the resulting feature map.

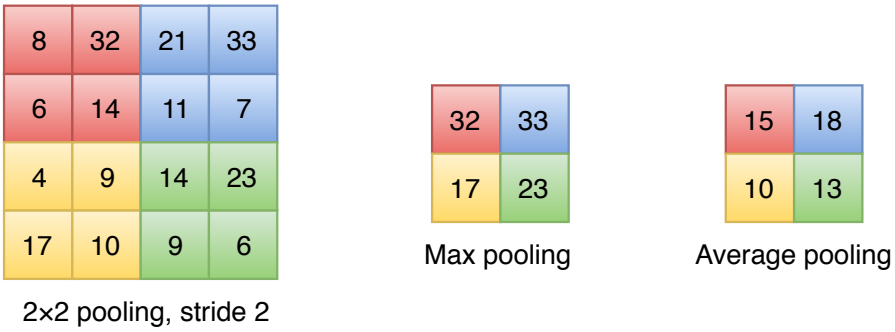


Figure 2.2: Various types of pooling operations. The max pooling function computes the maximum in the neighborhood of the window patches of size 2×2 with stride of 2. The average pooling takes average of the input elements in the window patch of size 2×2 with stride of 2.

Fully-Connected Layers Fully-connected ANN layers are typically used in the final stages of a CNN for classification or regression based on feature maps obtained through the convolutional filters. The output vector is then passed to a softmax function to obtain classification scores.

2.2 Systematic Development of 1D vs. 2D vs. 3D CNNs and Comparison to Other Approaches

Hyperspectral image can visually described as a three-dimensional data cube with spectral sampling along the z-axis. CNN can then be categorized into three groups: (a) 1D CNNs extracting spectral features, (b) 2D CNNs extracting spatial features, and (c) 3D CNNs extracting combined spectral-spatial features.

2.2.1 1D CNNs

A 1D CNN is responsible for pixel-level extraction of spectral features. As it is shown in Fig. 2.3, a hyperspectral vector is sent to the input layer and propagates through successive convolutional and pooling layers for feature extraction. Each convolutional layer has multiple convolutional filters, with sizes set using a hyperparameter. The output featuremap of each convolutional layer is a 1D vector. 1D CNNs have been applied for multiclass pixel-level classification of HSI data[19]. We note that although such 1-D CNNs can be applied to pixel level spectral reflectance data, they are not expected to be nearly as powerful in extracting abstract deep features as CNNs that operate on spatial information. This is because the one of the key drivers in the successful application of CNNs to imagery data stems from the multi-layer spatial convolutions that result in abstract deep spatial features representing the object morphology. Along a single dimension (a spectral reflectance profile of a pixel, for e.g.), there are no such features of interest to learn (e.g. features representing edges or texture). At best, such 1-D CNNs can then better condition (e.g. through the series of filtering layers) the spectral reflectance data to make it more robust to variations before it is classified.

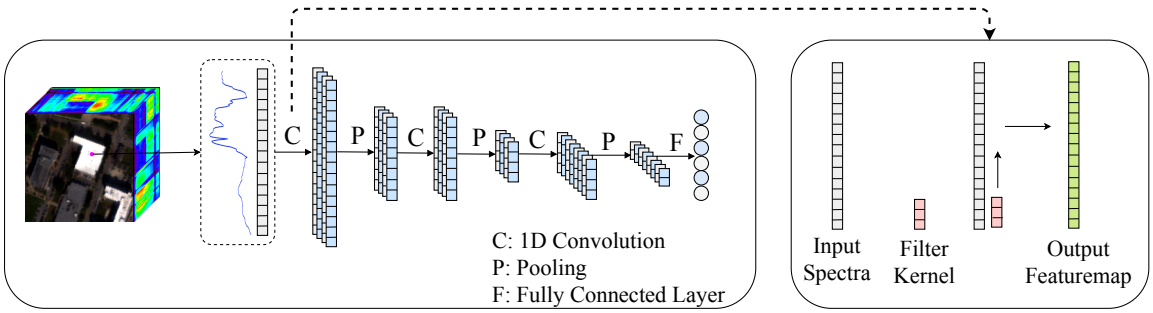


Figure 2.3: An architecture of 1D Convolutional Neural Network for Hyperspectral images; A typical 1D CNN consist of 1D convolutional layers, pooling layers, and fully connected layers. In 1D CNNs, filter kernel is 1D which convolve the hyperspectral cube in spectral dimension.

2.2.2 2D CNNs

Initial work on 2D CNNs (Fig. 2.4) made use of 2D CNN networks for classification of HSI data by taking a neighborhood window of size $w \times w$ around each labeled pixel and treat the whole window as a training sample [20, 21] to extract spatial features. Applying 2D CNN naively to HSI produces a feature map for each band. Since HSI data is often composed of several bands, this produces a large number of parameters that increase overfitting and computational requirements. Subspace learning methods are employed to reduce the spectral dimensionality prior to 2D feature extraction. Unsupervised methods such as principle component analysis (PCA) [27, 20, 21] have been exploited to reduce spectral dimensionality a practicable scale before 2D CNN training. However, the separate extraction of spectral and spatial features do not completely utilize spectral-spatial correlations within the data.

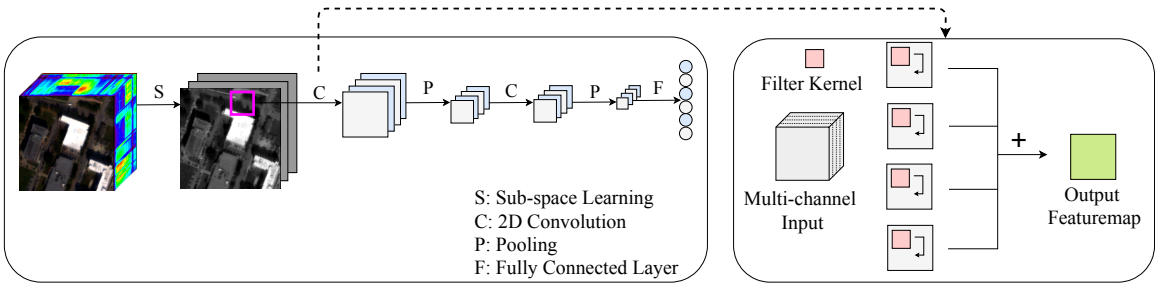


Figure 2.4: An architecture of 2D Convolutional Neural Network for Hyperspectral images; A typical 2D CNN consist of 2D convolutional layers, pooling layers, and fully connected layers. In 2D CNNs, filter kernel is 2D which convolve the hyperspectral cube in spatial dimension.

2.2.3 3D CNNs

After 1D CNN and 2D CNN which extract spectral features and local spatial features of each pixel, respectively, 3D CNN (Fig. 2.5) was introduced to learn the local signal changes in both the spatial and the spectral dimension of the HSI data, and exploit important discrimination information. 3D CNN model takes advantage

of the structural characteristics of the 3D HSI data and can exploit the joint spectral-spatial correlations information because the 3D convolution operation convolves the input data in both the spatial dimension and the spectral dimension simultaneously, while the 2D convolution operation convolves the input data in the spatial dimension. For the 2D convolution operation, regardless of whether it is applied to 2D data or 3D data, its output is 2D, while for 3D convolution operation, its output is also a cube. 3D CNN network has been investigated to learn rich spectral-spatial information for hyperspectral data classification [28].

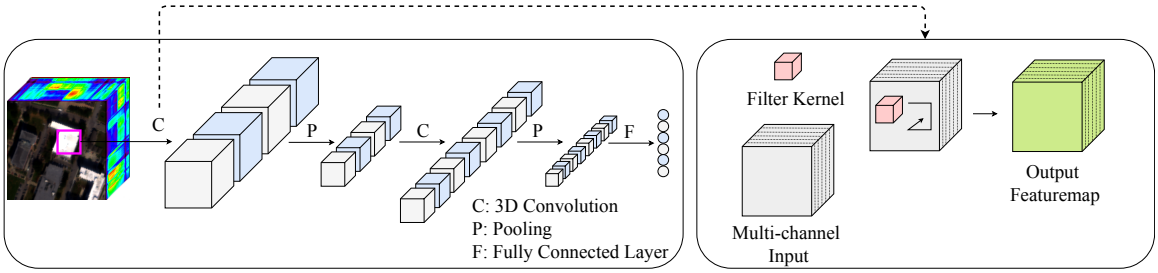


Figure 2.5: An architecture of 3D Convolutional Neural Network for Hyperspectral images; A typical 3D CNN consist of 3D convolutional layers, pooling layers, and fully connected layers. In 3D CNNs, filter kernel is 3D which convolve the hyperspectral cube in both spatial and spectral dimensions.

2.2.4 CNNs with RNNs (CRNNs)

A hybrid of convolutional and recurrent neural networks so-called CRNN (Convolutional Recurrent Neural Network) [29, 30] is composed of several 1D convolutional and pooling layers followed by a few recurrent layers, as it is shown in Fig. 2.6. CRNN has the advantages of both convolutional and recurrent networks. First, the 1D convolutional layers are exploit to extract middle-level locally invariant features from the spectral sequence of the input. Second, the recurrent layers are used to obtain contextual information from the feature sequence obtained by the previous 1D CNN. Contextual information captures the dependencies between different bands in the hyperspectral sequence, which is useful for classification task. For the recurrent layers,

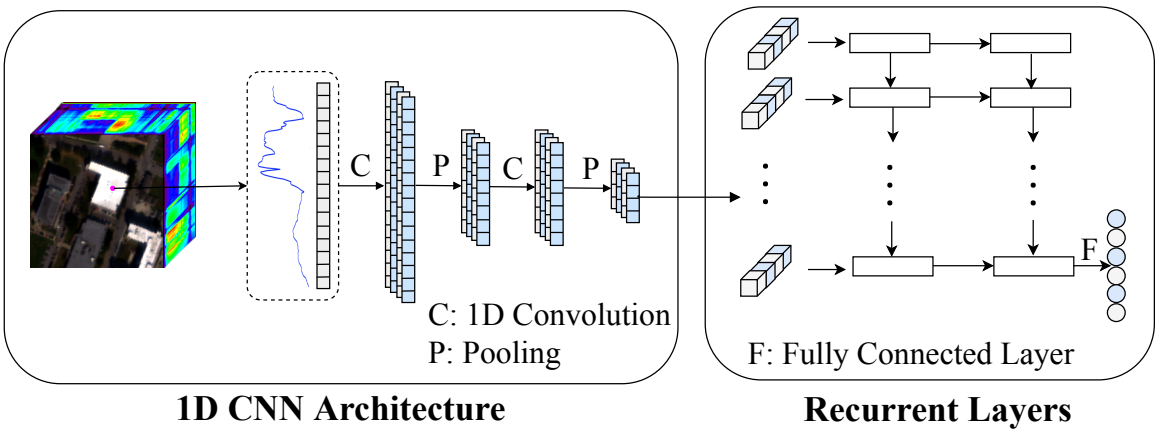


Figure 2.6: An architecture of Convolutional Recurrent Neural Network for Hyperspectral images; First part: 1D CNN is exploit to extract middle-level locally invariant features from the spectral sequence of the input. Second part: The recurrent layers are used to obtain contextual information from the feature sequence obtained by the previous 1D CNN.

the regular recurrent function or LSTM, which can capture very long dependencies, can be used. For cases with long length hyperspectral sequence which have long-term dependency, LSTM can be applied. At the end of this model, as in RNN, the last hidden state of the last recurrent layer will be fully connected to the classification layer. For training, as in CNN and RNN, the loss function is chosen as cross-entropy, and mini-batch gradient descent is used to find the best parameters of the network. The gradients in the CNN part are calculated by the back-propagation algorithm, and gradients in the RNN part are calculated by the back-propagation through time (BPTT) algorithm[31]. CRNN have been used to learn discriminative features for hyperspectral data classification[32].

We note that an interesting variation of this idea would be a hybrid model where the first part of the network extracts spatial features from the data through per-channel (spatial) convolutional layers, and the latter part of the network models the evolution of the spectral envelope through a recurrent network. Although we study pixel-level use of recurrent networks to hyperspectral data (modeling spectral reflectance/absorbance evolution) in these chapters, modeling both spatial and spec-

tral information will enhance this idea significantly by jointly leveraging spatial and spectral information.

2.3 Affect of Machine Learning Performance on Training/Testing biases – e.g., Random vs. Disjoint Datasets

Hyperspectral data are often large images (both due to their spatial dimensions and the number of spectral channels per pixel). For supervised learning, it is essential to acquire labeled training data (pixels or frames) to learn the classification model, and labeled test data (pixels or frames) to validate the classification models before they are deployed. When constructing such libraries of labeled samples, attention must be paid to the correlation within and between training and testing samples, the size of the labeled data pool and the mechanism through which labeled frames are extracted from large images.

- **Creation of training and testing datasets** - Ideal training and test sets are completely disjoint to ensure that there is no bias in reported accuracy. In many remote sensing tasks, a common approaches extract patches from a single large image – if training and test patches overlap, results may not be representative. Consequently, one may get an incomplete picture of the classifier's ability to generalize to new data. This may be unavoidable for some applications, but efforts should be made to minimize overlap between frames. In many works in the remote sensing community, it is customary to draw training and test samples (e.g. frames) randomly from the imagery/scene – in this setup, care must be taken to minimize or eliminate overlap between training and test frames that result. Remote sensing HSI data are also affected by clouds and other factors, such as sun-sensor-object geometry, variations in illumination, atmospheric conditions, and viewpoints. When constructing labeled libraries,

efforts should be made to ensure this variability is represented in the training pool.

- **Extracting window patches/frames from HSI data for image-based classification** - When selecting spatial patch size, it is critical to account for the image resolution with respect to object sizes. If the spatial resolution is coarse, individual frames may contain multiple classes. If a large part of the resulting frame contains class information other than the class of the center pixel, it may not result in features that represent that class, but instead represent the background. When utilizing frames with 3 dimensional filters (such as in 3D CNNs), care should be chosen when setting the frame size – too large a frame size relative to the size of objects will result in the spectral direction of the spatial-spectral filters learn from heavily mixed spectra in the frame. Once a frame size is fixed, one way to further mitigate this issue is to apply a threshold on the window patch based on occupation (i.e., based on the dominant class in that frame). For example, at a threshold of 50%, we require that at least 50% of pixels in the frame are from the class belonging to the central pixel, otherwise, we do not include that frame in the dataset. The size of window patches is related to the resolution of the HSI image, and the size of convolutional filter should be chosen based on the size of the window patches.

- **Size of labeled samples** - Sample size plays a key role in deep learning applications. Deep learning networks such as CNNs often require a large number of labeled samples to ensure effective learning (and convergence) of the network parameters. Remote sensing data processing has been challenging because ground truth is often limited, often difficult and expensive to acquire. However, there exists abundant unlabeled data that can be leveraged as part of the training. Addressing this problem is an area of active research exploration within the ar-

eas of data augmentation [33], semi-supervised data analysis [34], and domain adaption [35]. These are discussed in Chapter 5.

- **Disjoint correlations across bands in absorbance spectra** - Chemical compounds are composed of molecular bonds represented at widely varying absorbance bands. Correlations between peaks in the spectral signature are therefore minimally dependent on distance in the spectrum. As a result, convolutional filters trained in the spectral dimension are of limited use in infrared spectroscopy. However, dimensionality reduction can be useful for capturing these correlations.

2.4 Application to HSI for Remote Sensing

Deep learning is now being deployed for a variety of remotely sensed image analysis tasks. In [36], authors introduced a deep learning-based unsupervised feature extraction for hyperspectral data classification using autoencoders. They show that autoencoder extracted features increase accuracy of SVM and logistic regression backend classifiers and obtain better accuracy than conventional feature extraction such as PCA. One-dimensional CNNs have been used to leverage spectral features in remote sensing data [19]. In this method, due to limited number of training samples, a network with only one convolutional layer and one fully connected layer was used. Two-dimensional CNNs were applied to the first 10 to 30 principal components (applied on the per-pixel spectral reflectance feautures) of the hypercube to learn spectral and spatial properties of the HSI data for classification [27]. Recent studies have extracted spatial-spectral features using three-dimensional CNNs, such as [37] to analyze HSI data, where the authors proposed a dense convolutional network that uses dilated convolutions[38] instead of scaling operations to learn features at different scales. Recurrent neural networks (RNNs) which are designed to handle sequential

data have also been investigated as a tool for pixel level analysis of spectral reflectance features [39, 40, 32]. Hyperspectral data are treated as spectral sequences, and an RNN is used to model the dependencies between different spectral bands. Sometimes, RNNs and CNNS are used together to make a robust pixel-level classification [32]. First, convolutional layers can extract middle-level, locally invariant features from the input sequence, and the following recurrent layers extract spectral-context. There is an alternate convolutional processing for data that resides on manifolds and graphs, and the resulting convolutional neural networks are referred to as graph-based convolutional neural networks (GCNs). In [1], the authors demonstrate a framework that can use GCNs to effectively represent data residing on smooth manifolds, such as reflectance spectra of hyperspectral image pixels. In GCNs, a convolution operator is defined based on the graph Fourier transform to perform convolution/aggregation operations on feature vectors of its neighbors on the graph. A key element to successfully deploy graph-based networks is construction of an effective affinity matrix. In this work, the authors proposed a semi-supervised affinity matrix construction that was able to leverage a few labeled samples along with a large quantity of unlabeled pixels.

2.5 Results and Experiments Remote Sensing and Biomedical Hyperspectral Images

2.5.1 Remote Sensing Experiments

Since the dataset is highly unbalanced (as our several datasets in remote sensing applications), to compare the classification performance of deep learning architectures for each class of the dataset, we run our experiments with the same number of training samples for each class. The results are shown in Table 2.1 and Table 2.2 for disjoint and random datasets, respectively. As can be seen Table 2.1, spatial

properties of the commercial class helps in classification and improves the accuracy significantly. From Table 2.2, in some classes such as residential, railway, and parking Lot 2, spatial proprieties provide discriminative information, hence favoring 2D and 3D CNNs. Compared to 1D CNN and 2D CNN which exploit spectral and spatial information of HSI data respectively, applying 3D CNN results in improved accuracy for grass-synthetic, tree, residual, commercial, road, and parking Lot1. This shows that exploiting both spatial and spectral information simultaneously can be helpful for classes that have distinct spectral and spatial properties. For most of the classes, CRNN achieved the better performance compared to 1D CNN which indicates that by combining convolutional and recurrent layers, CRNN model is able to extract more discriminative feature representations by exploiting dependencies between different spectral bands. From the CRNN performance, it can also be seen that the recurrent layers can extract the spectral dependencies more effectively from the middle-level features provided by convolutional layers.

Table 2.1: Per class classification accuracy % on deep learning models for disjoint and random dataset extracted from UH 2013 dataset. Disjoint dataset - 100 sample per class for train and 80 samples per class for test.

	Method				
class	1D CNN	2D CNN	3D CNN	CRNN	RNN
1-Grass-healthy	81.42	80	73.75	74.28	75.71
2-Grass-stressed	74.28	76.25	73.75	81.42	75.71
3-Grass-synthetic	100	88.75	97.5	100	100
4-Tree	57.14	90	90	88.57	90
5-Soil	82.85	95	73.75	87.14	58.57
6-Water	94.28	88.75	87.5	98.57	87.14
7-Residual	45.71	68.75	62.5	38.57	44.28
8-Commercial	2.85	58.75	58.75	4.28	10
9-Road	61.42	41.25	57.5	52.87	50
10-Highway	44.28	43.75	61.25	50	51.42
11-Railway	57.14	38.75	51.25	72.85	64.28
12-Parking Lot 1	1.42	15	15	18.57	21.42
13-Parking Lot 2	71.42	93.75	93.75	77.14	72.85
14-Tennis Court	98.57	96.25	98.75	100	100
15-Running Track	94.28	97.5	98.75	95.71	90
Average accuracy of all classes	64.47	71.5	72.91	69.33	66.1

Table 2.2: Per class classification accuracy % on deep learning models for disjoint and random dataset extracted from UH 2013 dataset. Random dataset - 200 sample per class for train and 80 samples per class for test.

	Method				
class	1D CNN	2D CNN	3D CNN	CRNN	RNN
1-Grass-healthy	91.42	100	100	98.57	97.14
2-Grass-stressed	98.57	98.75	98.75	97.14	92.85
3-Grass-synthetic	100	97.5	100	98.57	100
4-Tree	95.71	91.25	98.75	97.14	100
5-Soil	95.71	100	100	98.57	97.14
6-Water	97.14	100	93.75	95.71	94.28
7-Residual	78.57	93.75	95	90	78.57
8-Commercial	92.85	82.5	92.5	95.71	92.85
9-Road	85.71	81.25	96.25	88.57	84.28
10-Highway	77.14	92.5	86.25	90	92.85
11-Railway	77.14	90	90	87.14	87.14
12-Parking Lot 1	84.28	81.25	87.5	88.57	97.14
13-Parking Lot 2	55.71	96.25	92.5	70	77.14
14-Tennis Court	100	98.75	100	98.57	98.57
15-Running Track	95.71	100	100	98.57	97.14
Average accuracy of all classes	88.37	93.58	95.41	92.85	92.47

The classification maps when using the different classification approaches discussed in this chapter are shown in Fig. 2.7 and Fig. 2.8 for disjoint and random datasets respectively. The models used to generate these maps were trained using 200 samples per class and 100 samples per class for the random and disjoint datasets respectively. It can be seen in Fig. 2.7 that when disjoint training and test data are used, the models struggle to generalize, particularly because there are no representative training samples in the shadow region (c.f. Fig. 2.7). Black rectangles in in these maps highlight the classification performance in specific areas for different models where we want to highlight specific trends with respect to improved classification with CNN/CRNN and their variants.

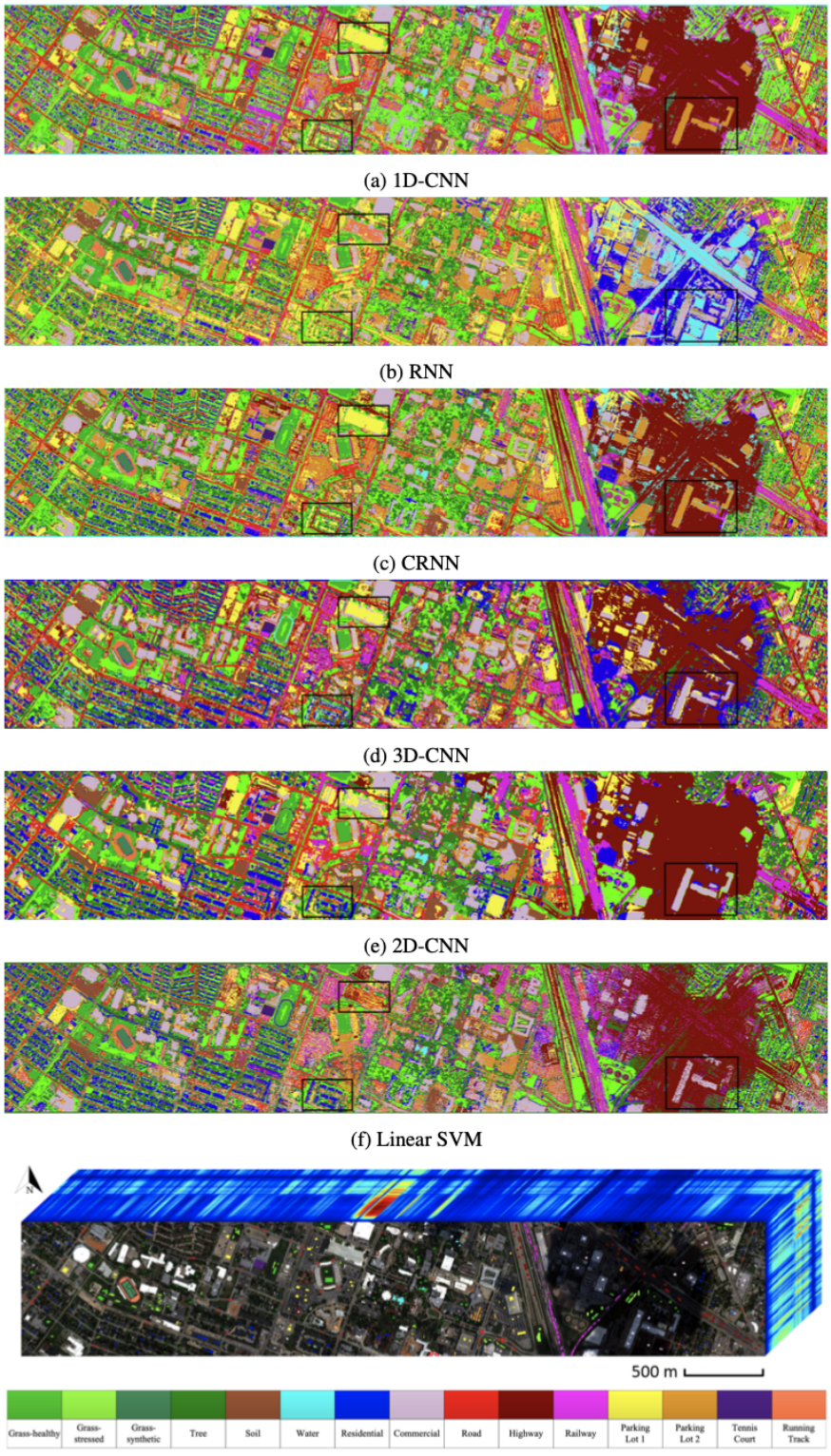


Figure 2.7: Classification maps of UH 2013 hyperspectral image computed from deep learning models trained by disjoint dataset extracted from UH 2013; All the models are trained using 100 samples per class.

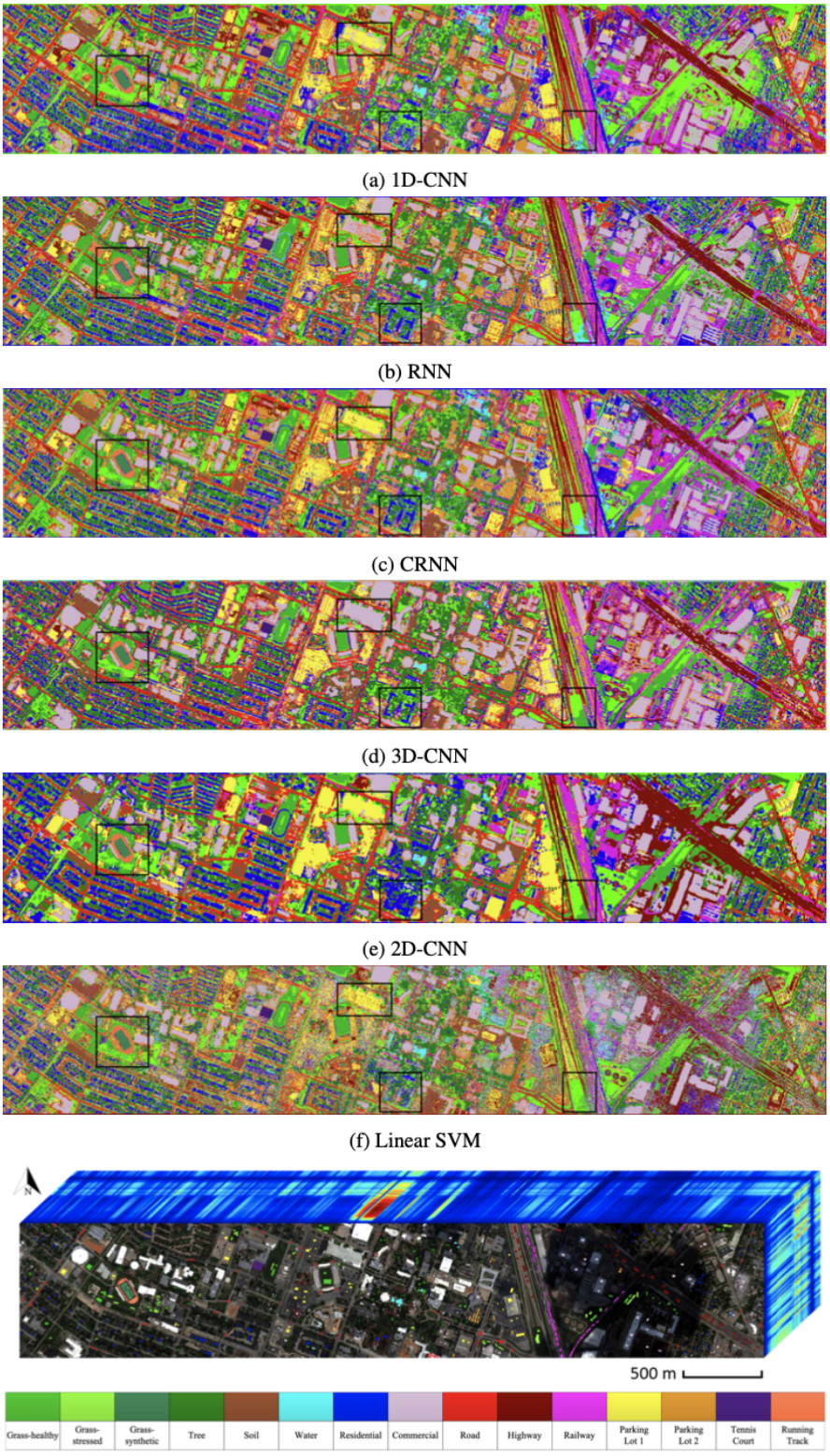


Figure 2.8: Classification maps of UH 2013 hyperspectral image computed from deep learning models trained by random dataset extracted from UH 2013; All the models are trained using 200 samples per class.

2.5.2 Biomedical FTIR Experiments

The classification performance of each network for different classes is studied in this experiment. The same number of training samples for each class is used to train different architectures. Three models per network are trained to compute the average accuracy for each class (Table 2.3). From Table 2.3, in some classes such as adipocytes and myofibroblasts provide strong spatial informationand 2D CNN works the best in these classes. For Collagen, epithelium, and necrosis, RNN shows the best accuracy. We can conclude that in these classes, there is a correlation between adjacent bands and RNN could capture this characteristic of the data very well.

Table 2.3: The average and standard deviation for per class accuracy for classification networks on the hypersperctral biomedical dataset are presented. The same number of training samples for all classes are used to train machine learning models and test their performances.

class	Method				
	1D CNN	2D CNN	3D CNN	CRNN	RNN
adipocytes	42.6	89.9	85.2	7.88	15.5
collagen	96.6	97.7	96.4	95.5	98.7
epithelium	88.5	89	85.6	90	91.2
myofibroblasts	64	90.9	74.5	64.2	62.4
necrosis	44	84.1	72.8	78.8	94.1
average accuracy	67.1	90.3	82.9	66.4	72.4

The classification maps for biomedical dataset are generated by trained models. The required memory limited us to compute the classification map for 3D CNN model. 5 classes are distinguished in 4 cores of the hyper spectral data using neural network (Fig. 2.9).

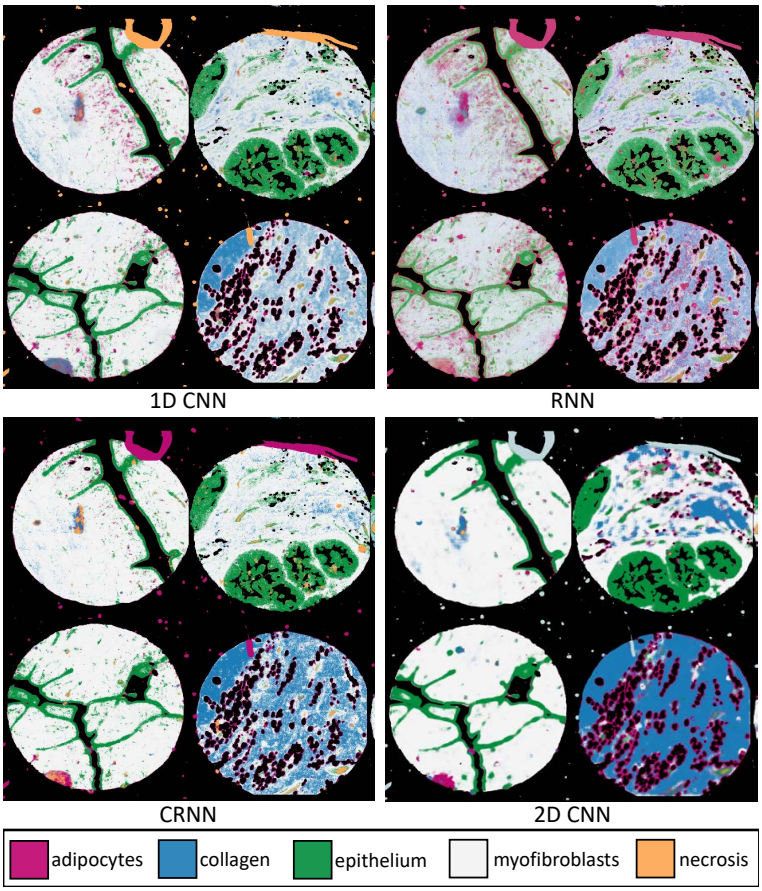


Figure 2.9: Different classes of the biomedical dataset are identified from the hyperspectral data by neural networks.

Chapter 3

Graph based Convolutional Neural Networks for hyperspectral image analysis

3.1 Why GCN Over CNN?

High spectral resolution and the contiguous bands of Hyperspectral Images (HSI) enable robust image analysis [41]. Due to the dense spectral sampling of HSI data, the associated spectral information in many adjacent bands is highly correlated, resulting in much lower intrinsic dimensions than the number of channels over which one images. Manifold learning approaches have been extensively investigated for HSI classification to – representing the underlying manifold and estimate geodesic distances [42]. Traditional manifold learning approaches such as Isometric feature mapping (ISOMAP) [41], Laplacian Eigenmaps[43], Locality Preserving Projection (LPP)[44], Locally Linear Embedding (LLE)[45], and Local Fisher Discriminant Analysis (LFDA) [46] have been very successful and have validated the hypothesis that spectral reflectance data indeed resides on manifolds [47]. The graph embedding framework [48] has been shown to be effective for representing high-dimensional data residing on manifolds. The graph embedding framework provides a manifold coordinate system in which the geodesic distance along the manifold is just the linear distance.

In addition to traditional manifold learning, graph based convolutional neural networks (GCNs) have been recently developed for applications on high-dimensional irregular domains represented by graphs, such as citation network classification [49, 50], social network node classification [51], predicting activity levels for various molecules

[52], molecular feature extraction [53], text categorization [52] and handwritten digits classification [54, 55]. Such approaches vary in the construction of convolutional filters which can broadly be categorized into spatial and spectral filters [48]. Spatial filters operate directly on the original graph and adjacency matrix whereas spectral filters operate on the spectrum of the graph-Laplacian. For a given node, GCN [49] performs convolution/aggregation operations on feature vectors of its neighbors where the convolution operator is defined based on the graph Fourier transform. Successive application of these operations convolve information across the K^{th} order neighborhood (i.e., embedding of a node depends on all the nodes that are at most K steps away), where K is the number of successive operations of convolutional layers in the neural network model. GCN applies the same operations across all nodes in the graph.

To our knowledge, prior graph based deep learning methods have not been applied for classification of data residing on smooth manifolds. In this paper, inspired by prior works that have demonstrated the value of graph based manifold learning for hyperspectral image analysis, we propose a graph-based CNN architecture for supervised pixel level classification of spectral signatures derived from hyperspectral imagery. Graph construction of the data forms a crucial step in the success of such algorithm, and hence we also propose robust semi-supervised approaches to construct the adjacency matrix.

3.2 Basic Algorithmic Description (Only Spectral GCN)

Fig. 4.6 illustrates the proposed framework for robust HSI classification - we contend that the key to successfully leveraging GCN spectral filters is to carefully construct affinity matrices in a way that leverages spatial-spectral properties of hyperspectral images. In this work, per-pixel HSI spectra goes through a CNN model

as a preprocessing step (extracting higher-level features from the spectral reflectance curves). These features are then passed on to a GCN. CNN here works as a data-learned filter to mitigate affects of noise and within-class variability on the reflectance spectra. We propose three possible ways to construct the adjacency matrix needed for constructing the spectral graph-based network: (a) A *supervised adjacency matrix* computed from discriminative features from a deep CNN, (b) an *unsupervised adjacency matrix* that uses the raw reflectance spectra to compute the adjacency matrix, and (c) a *semi-supervised adjacency matrix* that leverages information from true labels available in the training dataset and *pseudo-labels* (representing the intrinsic cluster structure) inferred from the entire dataset.

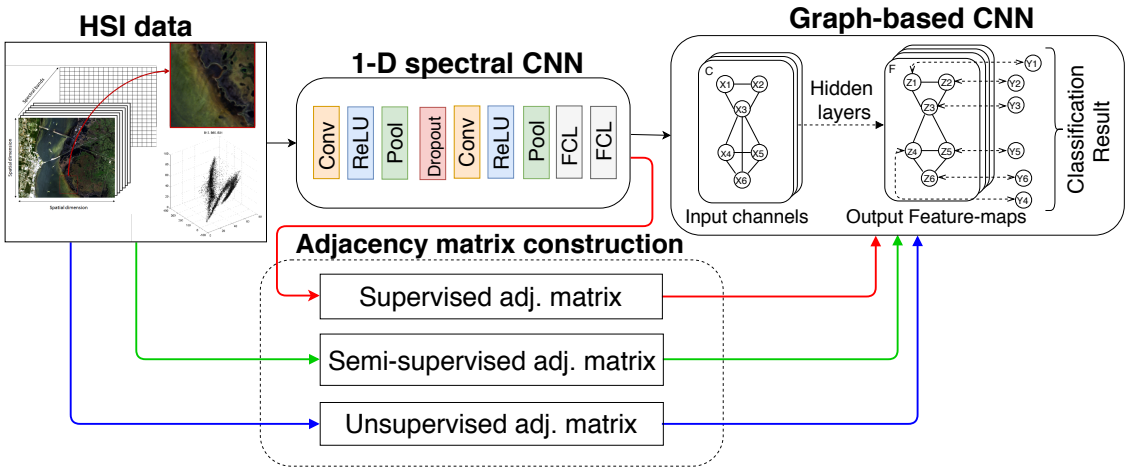


Figure 3.1: System Flowchart; three different adjacency matrix constructions; multi-layer Graph CNN for learning with C input channels and F feature-maps in the output layer; Labels are denoted by L_i.

3.3 Graph Convolutional Neural Network

3.3.1 Preliminaries of Graphs

An undirected graph is represented by $G = (\mathcal{N}, \mathcal{E})$, where $\mathcal{N}$ is the node set with $|\mathcal{N}| = N$ and $\mathcal{E}$ is the edge set. Denote by $A = [a_{ij}] \in \mathbb{R}^{N \times N}$ the adjacency matrix.

$D = diag(d_1, d_2, ..., d_N)$ is the degree matrix of A where $d_i = \sum_j a_{ij}$ is the degree of node i. The graph Laplacian [56] is defined as $L := D - A$, and the normalized graph Laplacian is defined as $\hat{L} := I_N - D^{\frac{-1}{2}} A D^{\frac{-1}{2}}$, where I_N is the identity matrix.

3.3.2 Graph Convolutional Neural Network (GCN)

Spectral Graph CNNs define the convolution by multiplication of a graph signal $x \in \mathbb{R}^N$ (a scalar for each node) with a spectral filter g_θ which is a function of eigenvalues of $\hat{L}$ [49]. However, this model requires computing the eigenvectors of the Laplacian matrix, which is impractical for large graphs. A way to circumvent this problem is by approximating the spectral filter g_θ with Chebyshev polynomials up to the K^{th} order [57]. Defferrard et al. [55] applied this to build a K-localized ChebNet, where the convolution is defined as

$$g_{\theta'} * x \approx \sum_{k=0}^{K} \theta'_k T_k(\tilde{L}) x \tag{3.1}$$

where $x \in \mathbb{R}^N$ is the signal on the graph, $g_{\theta'}$ is the spectral filter, $*$ denotes the convolution operator, T_k refers to the Chebyshev polynomials, $\tilde{L} = \frac{2}{\lambda_{max}} \hat{L} - I_N$, which λ_{max} denotes the largest eigenvalue of $\hat{L}$, and $\theta' \in \mathbb{R}^K$ is a vector of Chebyshev coefficients.

Kipf and Welling [49] limited $K = 1$ to simplify this model and approximated λ_{max} of $\hat{L}$ by 2. Under this approximation, the convolution becomes

$$g_\theta * x = \theta \left(I_N + D^{\frac{-1}{2}} A D^{\frac{-1}{2}} \right) x, \tag{3.2}$$

where θ is Chebyshev coefficient. They further introduced a normalization trick and applied it to the convolution matrix

$$I_N + D^{\frac{-1}{2}} A D^{\frac{-1}{2}} \rightarrow \tilde{D}^{\frac{-1}{2}} \tilde{A} \tilde{D}^{\frac{-1}{2}} \tag{3.3}$$

where $\tilde{A} = A + I_N$ and $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$. Generalizing the above definition of convolution to a graph signal with C input channels, i.e., $X \in \mathbb{R}^{N \times C}$ (each node is associated with a C-dimensional feature vector), the propagation rule of this simplified model is

$$H^{l+1} = \delta \left(\tilde{D}^{\frac{-1}{2}} \tilde{A} \tilde{D}^{\frac{-1}{2}} H^{(l)} W^{(l)} \right) \tag{3.4}$$

where $H^{(l)}$ is the matrix of activations in the layer l, and $H^{(0)} = X$, $W^{(l)}$ is the trainable weight matrix in layer l, δ refers to the activation function (e.g. $ReLU(.) = max(0, .)$). This model is called Graph Convolutional Networks (GCNs). The model used in [49] has two layers which apply a softmax classifier on the output features

$$Z = softmax \left(\hat{A} ReLU \left(\hat{A} X W^{(0)} \right) W^{(1)} \right) \tag{3.5}$$

where $\hat{A} = \tilde{D}^{\frac{-1}{2}} \tilde{A} \tilde{D}^{\frac{-1}{2}}$, $softmax(x_i) = \frac{exp(x_i)}{\sum_i exp(x_i)}$. The loss function is defined as the cross-entropy error over all labeled examples

$$L := - \sum_{l \in \mathcal{Y}_L} \sum_{f=1}^{F} Y_{lf} ln Z_{lf} \tag{3.6}$$

where $\mathcal{Y}_L$ is the set of node indices with labels, F is the number of nodes in the output layer, which is equal to the number of classes, and $Y \in \mathbb{R}^{|\mathcal{Y}_L| \times F}$ is the label indicator matrix. The weight parameters $W^{(0)} \in \mathbb{R}^{C \times H}$ and $W^{(1)} \in \mathbb{R}^{H \times F}$ are learned via gradient descent. The GCN model combines graph structures and node features in the convolution, features of unlabeled nodes are mixed with those of nearby labeled nodes and propagated graph through multiple layers. It was shown in[49] that GCNs outperformed many state-of-the-art methods on some benchmarks such as citation networks.

3.4 Affinity/Adjacency Matrix Computation/Design Spectrally and Spatially Weighted Affinities

For some tasks, such as citation graphs [58], the data resides on a graph structure. However, when using these ideas to learn the intrinsic manifold structure of data (e.g. per-pixel spectral reflectance data), a crucial element in leveraging a graph-based network is the appropriate construction of the adjacency matrix, A from the data [59]. To that end, we study three possible approaches to construct robust adjacency matrix that can be used with graph convolutional networks for hyperspectral image analysis, detailed in sec. 2.2.1 - 2.2.3.

3.4.1 Unsupervised Adjacency Matrix

An unsupervised spatial-spectral similarity graph (one that does not use label information) can be constructed as the product of spatial proximity and spectral similarity terms. For samples i and j, we can define

$$A_U(i,j) = \exp(\frac{-\left\|X_i - X_j\right\|^2}{\delta_X^2}) \cdot \exp(\frac{-\left\|d_i - d_j\right\|^2}{\delta_d^2}) \tag{3.7}$$

where d_i denotes the spatial coordinates of pixel i, and X_i denotes pixel spectral reflectance vector, where δ_d and δ_X are variance terms for the heat kernel that define the spatial and spectral neighborhood size. Such a matrix was applied successfully for traditional manifold learning [60], and is a good candidate in our framework in this paper as well, since it ensures that only spatially neighboring samples contribute to the estimation of spectral affinities in the feature space, mitigating affects of spatial variabilities.

3.4.2 Supervised Adjacency Matrix

In [58], a *supervised graph construction* was proposed wherein output features from a supervised network were used to construct the resulting affinity matrix, which was then used by backend spectral networks that the authors proposed. Such an approach can learn affinities in a feature space that is highly discriminative. We adopt this idea and study its benefits within our framework. Given a training set $\hat{X} \in \mathbb{R}^{\hat{N} \times C}$, we train a CNN with H hidden layers of weights $W_1, ..., W_H$, using standard ReLU activations, max pooling, and dropout. We then extract the output featuremap of the last layer of the network which is a fully connected layer for the entire dataset, $o \in \mathbb{R}^{N \times M_H}$, where M_H is the dimension of output feature in the last layer, and N is the number of samples in the entire dataset. We then define the *spatial-spectral supervised adjacency matrix* A_S for samples i and j as follows

$$A_S(i,j) = \exp(\frac{-\left\|o_i - o_j\right\|^2}{\delta_o^2}) \cdot \exp(\frac{-\left\|d_i - d_j\right\|^2}{\delta_d^2}) \tag{3.8}$$

where spectral features in eq. (7) are replaced by the output featuremap o.

3.4.3 Semi-supervised Adjacency Matrix

Finally, we propose a *semi-supervised* adjacency matrix that draws on information provided by the limited amount of labeled data and extensive unlabeled data available for classification. This is accomplished by leveraging the notion of pseudo-labels, the intrinsic clusters in the dataset. In this paper, we use a variational inference based Dirichlet Process Mixture Model (DPMM) [61] – it not only clusters the data under a Gaussian mixture model assumption, but also infers the intrinsic number of mixtures as part of the inference. We contend that true labels would often be a subset of pseudo-labels (i.e., pseudo-labels may represent previously unseen classes or new modes of existing classes). By learning a graph structure that is based on both true

labels and pseudo-labels, we have an affinity matrix that reflects the class-specific and cluster-specific structures of the data. Our proposed semi-supervised adjacency matrix is described below. Consider binary weights $w(i,j)$ between samples i and j

$$w(i,j) = \begin{cases} 1, & \text{if } i \in \mathbb{T}, j \in \mathbb{T}, y_i = y_j, \\ 0, & \text{if } i \in \mathbb{T}, j \in \mathbb{T}, y_i \neq y_j, \end{cases} \tag{3.9}$$

$$w(i,j) = \begin{cases} 1, & \text{if } i \in \mathbb{E}, j \in \mathbb{T}, \hat{y}_i = \hat{y}_j, \\ 0, & \text{if } i \in \mathbb{E}, j \in \mathbb{T}, \hat{y}_i \neq \hat{y}_j. \end{cases} \tag{3.10}$$

$$w(i,j) = \begin{cases} 1, & \text{if } i \in \mathbb{E}, j \in \mathbb{E}, \hat{y}_i = \hat{y}_j, \\ 0, & \text{if } i \in \mathbb{E}, j \in \mathbb{E}, \hat{y}_i \neq \hat{y}_j. \end{cases} \tag{3.11}$$

where $\mathbb{T}$ and $\mathbb{E}$ represent training and test datasets, respectively. Here y and $\hat{y}$ represent true labels and pseudo labels, respectively. We then construct our spatial-spectral adjacency matrix as follows

$$A_P(i,j) = w(i,j) \cdot \exp\left(\frac{-\left\|d_i - d_j\right\|^2}{\delta_d^2}\right). \tag{3.12}$$

For pairs of samples in the training set, we use class-membership to define locality in the feature space. For pairs of samples in the test set or test and training set, we use pseudo-labels (cluster membership) to define locality in the feature-space.

3.5 Experiments on Remote Sensing HSI Data

The training process of our one-dimensional CNN starts with the weights of all layers randomly initialized and the initial learning rate set to 10^{-5}. Network parameters are updated using the Adam algorithm [62] with a batch size of 128 over 4000 epochs. The 1-D CNN used for preprocessing has 2 convolutional layers, each with 32 filters, and the 1-D CNN implemented for comparison has 4 convolutional layers, with the first two layers having 32 filters and the last layers having 64 filters.

The GCN implemented in our work has 4 hidden layers with the first 3 layers having 64 units, and the last layer having 15 units, and dropout rate of 0.5 (first and last layer). For GCN, we also used Adam algorithm with a learning rate of 10^{-2} over 1000 epochs. To stabilize optimization, we add a residual connection [63] between the GCN layers when the network has more than one layer. We added two residual connections, one for layer 1 and layer 2, and one for layer 2 and layer 3.

Fig. 3.3 shows the following adjacency matrices – A_S, spectral featuremap part of A_S, and A_P. For comparison, the adjacency matrix derived from the full ground truth (over training and test samples) is also provided. The white lines demarcating the two blocks along the diagonal show the training and test affinities respectively. Off diagonal blocks show the affinities between test and train samples. Samples in these matrices are arranged sequentially from class 1 to 15 for both train and test sections of the matrices. In Fig. 3.3-b, we use the first half (supervised featuremaps) of Eq. 8 ($\exp(\frac{-\|o_i-o_j\|^2}{\delta_o^2})$) and we see that inter-class correlation between both training and testing samples dominate the matrix and can confuse the underlying classifier. When we incorporate a spatial prior as we did in eq. 8 (Fig. 3.3-c), we see an improved affinity matrix that resembles the ground-truth matrix. The proposed semi-supervised adjacency matrix, A_P (Fig. 3.3-d) resembles the ground-truth matrix even better.

Classification performance, as quantified by overall classification accuracy is reported in Table 3.1. The trends in performance are consistent with the discussions above and our observations from the adjacency matrices. We note that a supervised CNN as a preprocessing to GCN yields robust classification (wherein the CNN learns 1-dimensional filters applied to the spectra, to address noise and variability in the spectral reflectance due to factors such as sun-sensor-object geometry, cloud-shadows etc.). For reference, results with a traditional CNN applied to the same spectral features is also provided. A graph convolution on these CNN processed features is the most effective strategy to learn the underlying manifold structures. Additionally,

we see that the proposed semi-supervised adjacency matrix (A_P) provides the best performance.

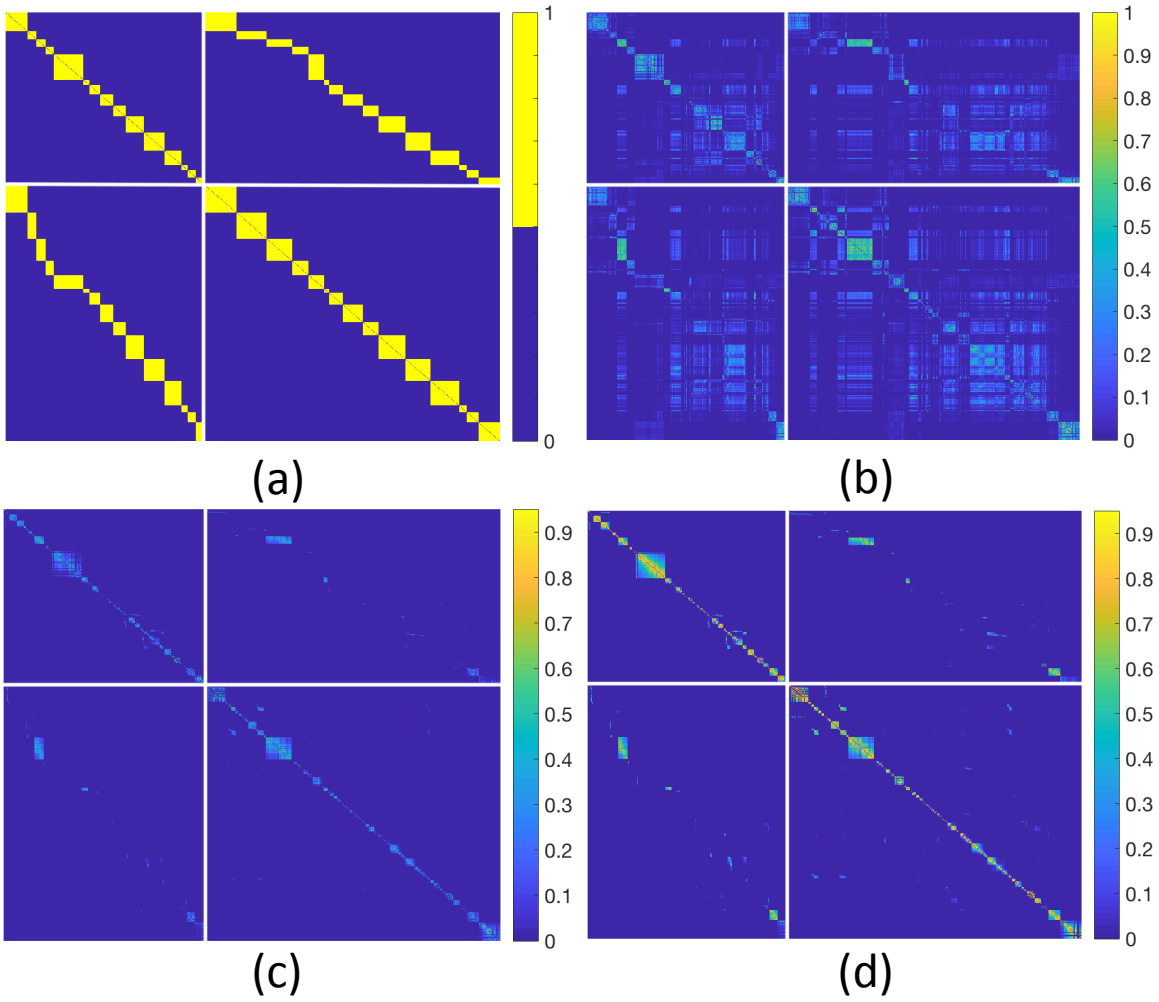


Figure 3.2: Adjacency matrices: a) ground truth, b) supervised (spectral featuremaps), c) supervised (spatial-spectral featuremaps), d) semi-supervised (spatial-spectral).

Table 3.1: Overall classification accuracy (%) and standard deviation (in parenthesis) when using traditional 1-D CNN and GCN architectures

Method (adj matrix)	1D-CNN	GCN (A_U)	GCN (A_P)	CNN + GCN (A_S)	CNN + GCN (A_P)
Accuracy	65.7 (0.7)	32.8 (0.7)	52.6 (1.4)	68.4 (0.5)	**71.2 (1.0)**

3.6 Experiments on Biomedical FTIR Data

Dataset: We used the HD dataset[64] consisted of TMA BR961, which was imaged using the Agilent Stingray imaging system comprised of a 680-IR spectrometer coupled to a 620-IR imaging microscope with 0.62 numerical aperture averaged with 32 co-additions. The spectral resolution was $4cm^{-1}$ with a pixel size of $1.1\mu m$ and a truncated spectral range of 1000 to $3801cm^{-1}$. The dataset contained 96 cores from separate patients with cases of normal, hyperplasia, dysplasia and malignant tumors. The dataset contains 6 classes: (a) adipocytes, (b) blood, (c)epithelium, (d) collagen, and (e) necrosis. For this experiment, we try to classify all the classes except the blood class. For each class 500 samples randomly selected from the left cores of the image for the training part and 1000 samples selected form the right cores for the test purposes.

Experimental Setup and Results: The training process of our For one-dimensional CNN used for preprocessing, the weights are initialized Xaviar unifrom, and the initial learning rate set to 10^{-4}. Network parameters are updated using the Adam algorithm [62] with a batch size of 128 over 128 epochs. It has 2 convolutional layers, each with 32 filters. The two-dimensional CNN used to make supervised affinity matrix and pseudo-labels for semi-supervised affinity matrix, its weights are initialized Xaviar unifrom. Network parameters are updated using the Adam algorithm [62] with a batch size of 128 over 64 epochs with learning rate of 10^{-5}. The GCN implemented in our work has 4 hidden layers with the first 3 layers having 64 units, and the last layer having 15 units, and dropout rate of 0.5 (first and last layer). For GCN, we also used Adam algorithm with a learning rate of 10^{-2} over 1000 epochs. To stabilize optimization, we add a residual connection [63] between the GCN layers when the network has more than one layer. We added two residual connections, one for layer 1 and layer 2, and one for layer 2 and layer 3.

Fig. 3.3 shows the proposed adjacency matrices. For comparison, the adjacency matrix derived from the full ground truth (over training and test samples) is also provided. The white lines demarcating the two blocks along the diagonal show the training and test affinities respectively. Off diagonal blocks show the affinities between test and train samples. Samples in these matrices are arranged sequentially from class 1 to 5 for both train and test sections of the matrices. In Fig. 3.3-d, for unsupervised adj. matrix, we can see that inter-class correlation between both training and testing samples dominate the matrix and can confuse the underlying classifier. The supervised adj. matrix shown in Fig. 3.3-c shows less noises and inter-class correlation between samples compared to unsupervised adj.matrix. The proposed semi-supervised adjacency matrix, A_P (Fig. 3.3-b) resembles the ground-truth matrix even better. However, some inter-class correlation can be also seen in it. The table 3.4 represents the number of clusters extracted by vi-DPMM method for psuedo-label construction and Normalized Mutual Information(NMI) which demonstrate the clustering performance.

Classification performance, as quantified by per class and overall classification accuracy for graph-based CNNs, and other deep learning models is reported in Table 3.2. The trends in performance are consistent with our observations from the adjacency matrices. For reference, results with 1D CNN, RNN, and CRNN applied to the same spectral features is also provided. A 2D CNN is also applied on $33 \times 33 \times 16$ window patches extracted from the dataset after applying PCA analysis. We note that a supervised CNN as a preprocessing to GCN yields robust classification (wherein the CNN learns 1-dimensional filters applied to the spectra, to address noise and variability in the spectral reflectance). For adipocytes class, GCN method with semisupervised adj. matrix results in the best accuracy performance. Classification results for myofibroblasts with 2D CNN shows that this class carries a good spatial information which can be captured by a spatial-based deep learning method. Ta-

Table 3.2: Per class classification accuracy % and standard deviation (in parenthesis) on deep learning and Graph based models.

	Method (adj matrix)							
class	1D CNN	2D CNN	CRNN	RNN	GCN(A_U)	GCN(A_S)	CNN+GCN(A_S)	CNN+GCN(A_P)
adipocytes	28.6	72.2	10.3	14.7	52.6	82.3	66	90.7
collagen	98	89.3	94.7	99.2	95	96.7	98.8	90.2
epithelium	87	89.5	88.8	88.4	83	84.8	89.2	89.2
myofibroblasts	66	77.3	71.3	65.5	53.3	25.7	73.1	61
necrosis	39.3	85.7	29.6	92.7	64.3	84.5	93.4	93.8
Average accuracy	63.8 (0.10)	82.8 (2.64)	58.9 (2.35)	72.1 (0.17)	69.6 (3.84)	74.8 (0.01)	84.1 (0.56)	85.1 (0.07)

ble 3.2 show that RNN can capture the dependencies between different bands in the hyperspectral sequence of collagen class very well and result in the best classification performance for this class. A graph convolution on these CNN processed features is the most effective strategy to learn the underlying manifold structures. Additionally, we can see that the proposed semi-supervised adjacency matrix (A_P) provides the best overall performance compared to other graph-based models.

Table 3.3 provides classification results based on some traditional machine learning techniques. As it is shown, SVM provides the best classification results for myofibroblasts class. Naive Bayes algorithm shows a significant improvement in classification performance of collagen and necrosis classes.

Fig 3.4 provides classification map for four cores of the dataset using deep learning models used in this work.

Table 3.3: Experimental classification results for various traditional machine learning models.

	Method				
class	SVM	Decision Tree	Random Forest	Naive Bayes	kNN
adipocytes	29.6	16.2	33	0	24.9
collagen	95.6	95.6	97.6	99.8	96.8
epithelium	84.3	79.9	84.9	0	82.8
myofibroblasts	91.4	65.4	61.2	27.6	60.9
necrosis	81.1	31.6	27.6	98.8	56.5
average accuracy	76.4	57.74	60.8	45.2	64.38

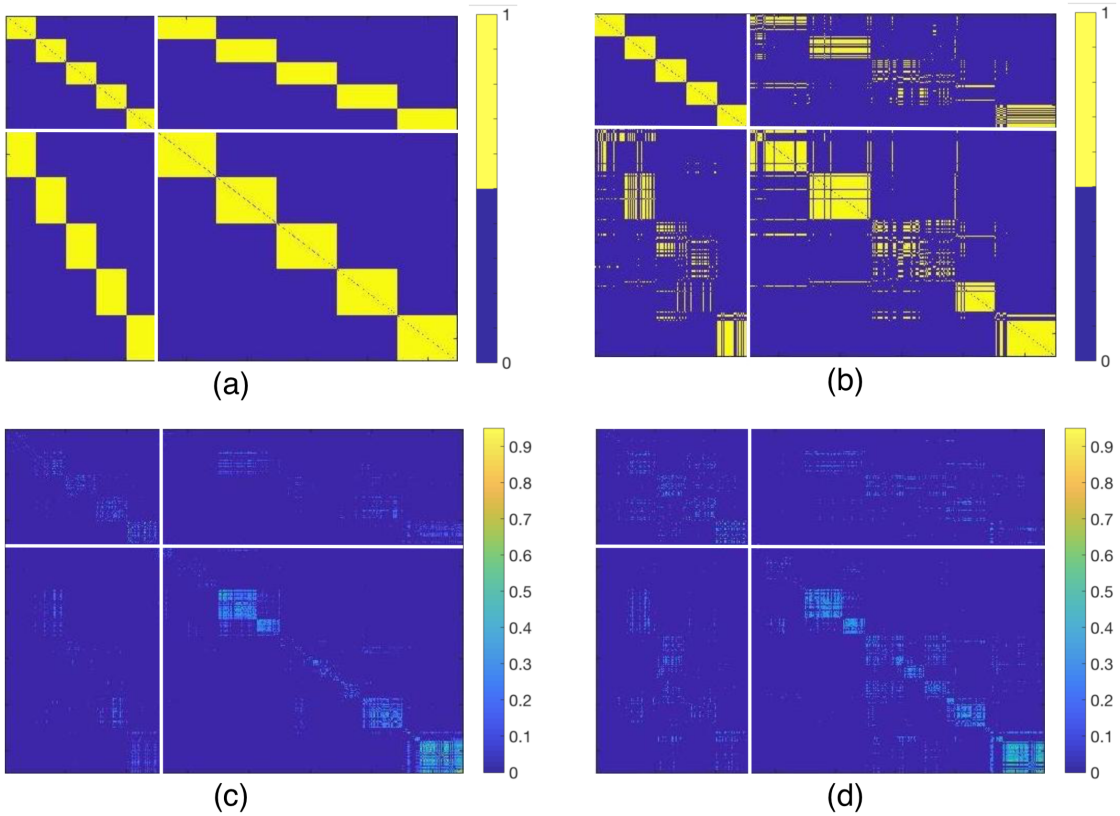


Figure 3.3: Adjacency matrices: a) ground truth, b) semi-supervised (A_P), c) supervised (A_S) , d) unsupervised (A_U).

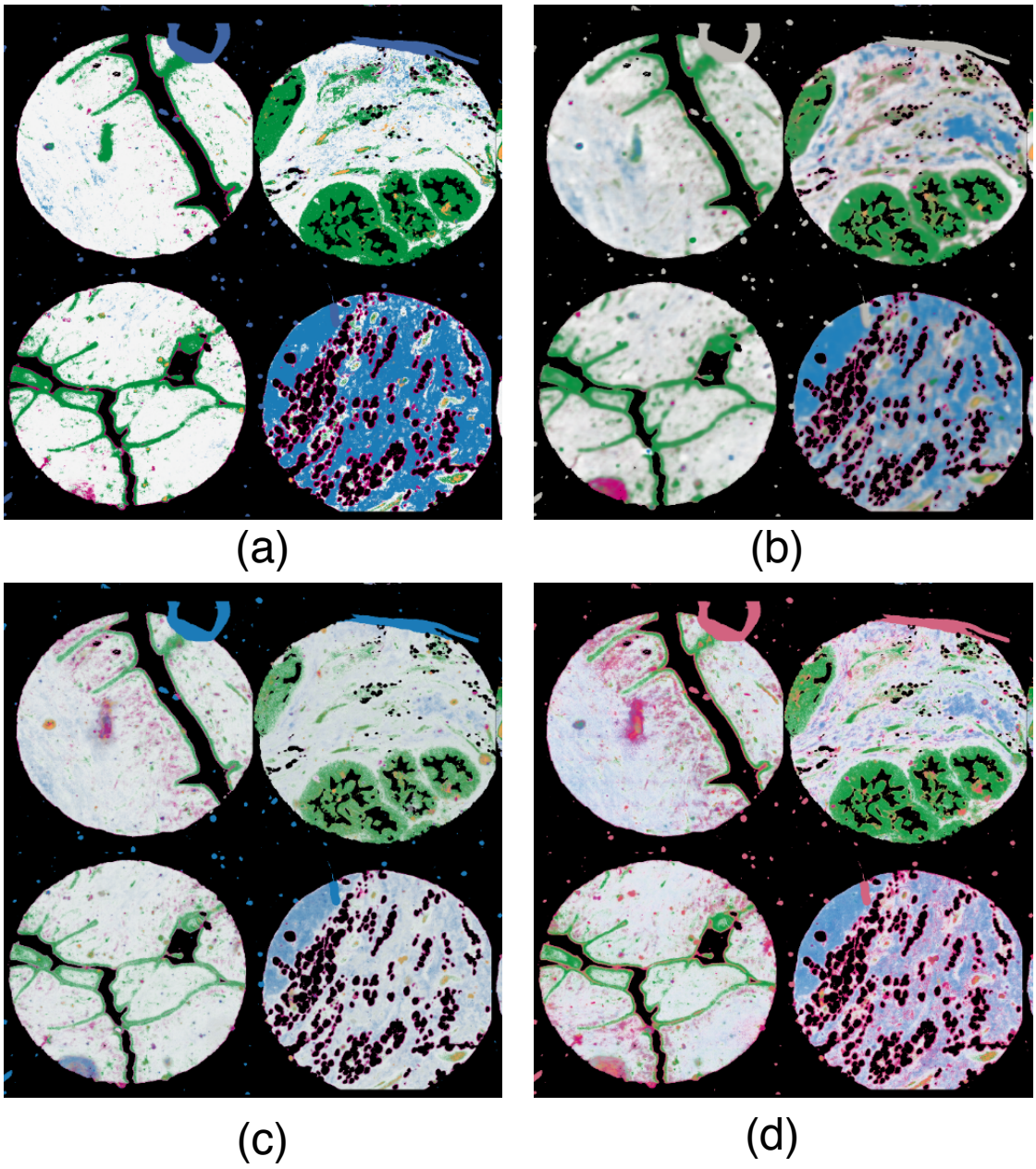


Figure 3.4: Classification maps for various deep learning models: a) CRNN, b) 2D CNN, c) 1D CNN, d) RNN.

Table 3.4: Clustering performance of DPMM used for Pseudo label

Number of clusters	NMI
20	60.57(1.2)

3.7 Joint Spatial and Graph Convolutional Neural Networks - A Hybrid Model for Spatial-Spectral Geospatial Image Analysis

Model for Spatial-Spectral Geospatial Image Analysis We also develop a fusion approach for hyperspectral image classification that leverages both spectral and spatial information in the data through GCNs and 2D CNNs respectively. We present results with a decision fusion approach that combines the posterior probabilities from each network, and a feature fusion approach that fuses these two networks into a single two-stream network wherein the GCN and 2D CNN features are fused prior to classification. Results with a hyperspectral image classification task demonstrate the efficacy of the proposed approach.

3.7.1 Feature and Decision Fusion

Decision Fusion - The predictions from the two base neural networks trained on the data source independently are then fused. In this paper, the GCN model is trained on the HSI pixels and the 2D CNN is trained on the corresponding window patches extracted from HSI image as is shown in Fig. 3.5-(left). In this method, after calculating the predictive probabilities for both networks: $P_C(y = k|\mathbf{x})$ and $P_G(y = k|\mathbf{x})$, the final predictive probability is computed using decision fusion, such as Linear Opinion Pooling (LOP)

$$P(y = k|\mathbf{x}) = w_1 P_C(y = k|\mathbf{x}) + w_2 P_G(y = k|\mathbf{x}) \tag{3.13}$$

where w_1 and w_2 are positive weights which satisfy $w_1 + w_2 = 1$.

Class-specific Decision Fusion - The decision fusion can be adaptive such that the strengths/weakness of each network specific to each class is leveraged – we refer to this as class-specific decision fusion. Here, the weights for each class are chosen based on per class training/validation accuracy of GCN and 2D CNN models respectively.

Deep Feature Fusion - As shown in Fig. 3.5-(right), in the deep feature fusion approach, we have one 2D convolutional sub-network and one graph convolutional sub-network designed to train image-level and pixel-level hyperspectral data, respectively. The deep features extracted by each network are concatenated before being fed to the output layer for classification. Following this, the concatenated features are used to calculate the cross entropy loss function for the fusion network. In the decision fusion model, two networks are trained independently. However, in the deep feature fusion network which is an end-to-end single classification model, the two sub-networks are trained simultaneously, and they benefit from each other.

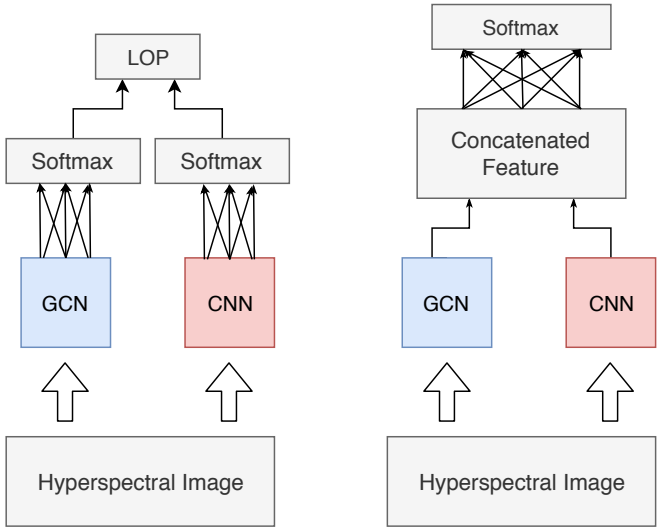


Figure 3.5: Fusion Networks - (left) Decision Fusion - (right) Feature Fusion.

3.7.2 Experiment Setup and Results

Fig. 3.6-a and Fig. 3.6-b represents the $w(i,j)$ part of A_P which is based on pseudo-labels computed from feature-maps of 1D and 2D CNNs, respectively. Fig. 3.6-c and Fig. 3.6-d represents the spectral part of A_S ($\exp(\frac{-\|o_i-o_j\|^2}{\delta_o^2})$) computed from feature-maps of 1D and 2D CNNs, respectively. As is shown in Fig. 3.6, the spectral parts of A_P (3.6-a) and A_S (3.6-c) computed based on 2D CNN feature-map demon-

strate less noise and inter-class correlation between testing and training data samples compared to those computed based on 1D CNN feature-map (3.6-b and 3.6-d). To mitigate the noise, the spectral parts of both supervised and semi-supervised matrices integrate with the spatial part as represented in eq. 9 and eq. 13, and shown in Fig. 3.6-e for semi-supervised adjacency matrix computed from 2D feature-maps. Fig. 3.6-f shows the ground truth adjacency matrix computed based on the true labels of the entire dataset. Diagonal blocks represented by white lines show the training and testing affinities, respectively. Off diagonal blocks represent the affinities between training and testing samples. For both train and test section of the matrices, classes are arranged sequentially from 1 to 15.

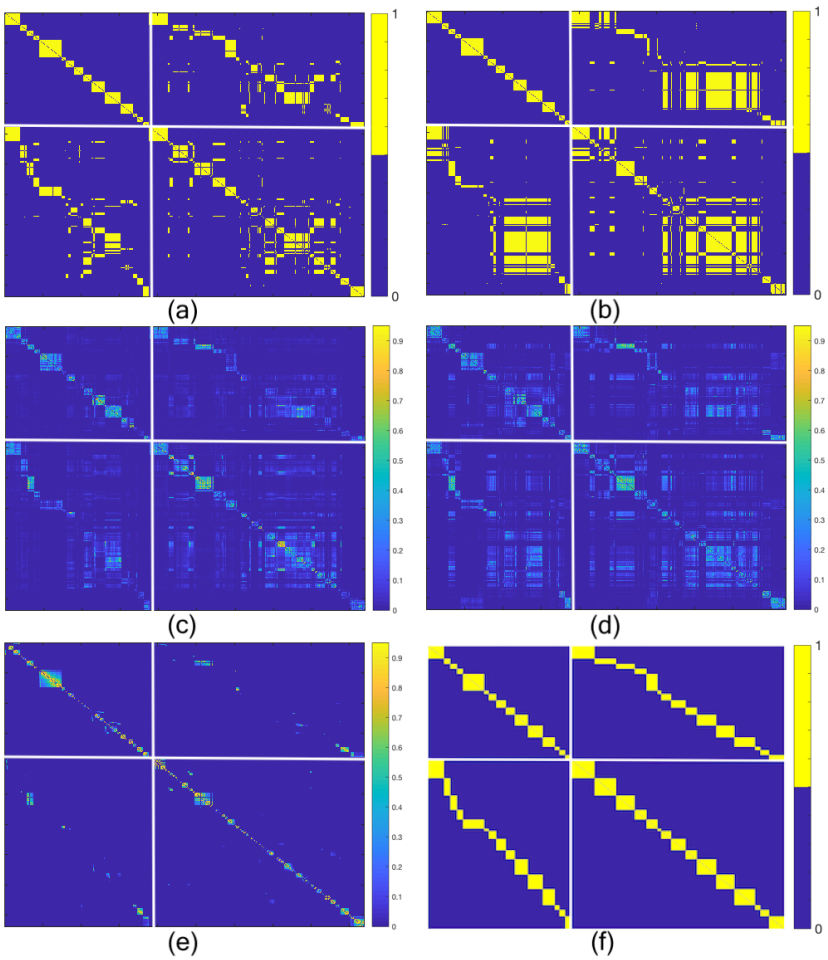


Figure 3.6: Adjacency matrices: a) semi-supervised (spectral) computed from 2D feature-map, b) semi-supervised (spectral) computed from 1D feature-map [1], c) supervised (spectral) computed from 2D feature-map , d) supervised (spectral) computed from 1D feature-map [1], e) semi-supervised (spectral-spatial) computed from 2D feature-map, f) ground truth.

In our experiments, the classification results for deep learning models and proposed fusion networks are presented in Table 3.5 where each experiment is repeated 3 times, and the average accuracy and the standard deviation is reported. For the fusion models, and the GCN model, we used supervised (A_S) and semi-supervised (A_P) adjacency matrix constructed based on 2D CNN feature-map. The 1D convolutional layers used as preprocessing the HSI data before being fed to GCN has two convolutional layers with each of them having 32 filters. 2D CNN used for adjacency matrix construction has 2 convolutional layers, the first layer has 32 filters and the last

layer has 64 filters. The 2D CNN implemented as a part of fusion network contains 3 convolutional layers with 32, 32, and 64 filters, respectively.

The GCN part of the fusion network contains 4 hidden layers with the first 3 layers having 64 units, and the last layer having 15 units. For 2D CNNs and GCN, we used Adam algorithm [65] for optimization with a learning rate of 10^{-4} and 10^{-2}, respectively. We add a residual connection [63] between the GCN layers when the network has more than one layer to stabilize optimization. We added two residual connections, one for layer 1 and layer 2, and one for layer 2 and layer 3. For feature fusion network, 2D CNN and GCN models are pretrained separately, and then the fusion model is fine-tuned to speed up the training process and fix the problem of different convergence time of GCN and 2D CNN. Based on the provided accuracies, we note that both decision fusion and feature fusion showed a better performance than traditional CNN models, even 3D CNN which is a good baseline for analysis of spectral-spatial features of HSI data. Experiments were carried out on a workstation with a 3.0-GHz Intel(R) Core i7-5960X CPU and an NVIDIA(R) GeForce Titan X GPU. Average training time per epoch for 3D CNN, GCN(A_P), Decision Fusion (A_P), and Feature Fusion (A_P) was $3s$, $6.4s$, $6.5s$, and $13.9s$ respectively.

Table 3.5: Overall classification accuracy (%) and standard deviation (in parenthesis) for 1D CNN, 2D CNN, 3D CNN, GCN, and fusion networks

Method	GCN		Decision Fusion		Decision Fusion (Class_specific)		Feature Fusion		1D CNN	2D CNN	3D CNN
Adj matrix	A_S	A_P	A_S	A_P	A_S	A_P	A_S	A_P			
Accuracy	68.56 (2.7)	71.85 (0.02)	79.65 (0.4)	79.46 (0.9)	82.19 (1.1)	**83.48 (0.7)**	78.87 (0.06)	82.28 (1.05)	66.3 (0.7)	78.2 (0.8)	70.74 (0.008)

Chapter 4

Atlas Fitting

4.1 Introduction to Atlas Fitting

Anatomical atlases in standard coordinate systems are essential to interpret and integrate of research findings in a common spatial context. For rat and mouse brain analysis, scientifics use these anatomical atlases to determine the locations of various cell types and neuronal circuits in each region of the brain for better understanding of brain functions. However, due to nonuniform and heterogeneous brain structure characteristics captured by imaging techniques, it is difficult to apply a single standard atlas for registration of various brain images. To overcome this issue, there are various manual and semi-manual mapping and constructing anatomical atlases introduced recently. In this work , they present a method, called DeepMapi [66], which is based on a convolutional neural network (CNN) to predict the deformation field corresponding to each pair of images and used to automatically register mesoscopic micro-optical imaging datasets to a reference atlas. The other registration work called BIRDS [67] is semi-automatic feature-based fitting work for the mapping and analysis of 3D microscopy data applied to the mouse brain. They combined the image registration method with a deep-learning algorithm which uses pixels in the image to identify features in isolated regions of the brain. In other study , a Brain Spatial Mapping Interface (BrainsMapi) [68] was proposed to map three-dimensional brain image datasets at the cellular level to the standard Allen CCFv3 brain atlas. There is another method called DeepBrainSeg [69], which combines registration and segmentation methods. They solve the issue of brain region segmentation for micro-optical

images based on a CNN, and to get a better result they used registration to locate brain regions before predicting the segmentation result to avoid over-segmentation and to improve efficiency. All of the above work used feature-based registration methods to map the brain images to a specific atlas with the same data type. However, there is no method to map the Paxinos binary atlas to the brain images and locate different cell types in brain regions. In this work, we combine image registration and segmentation methodologies to build a semi-automatic atlas fitting to aim rat brain image parsing. We have a comprehensive panel of biomarkers from serial whole-brain slices, characterizing all major brain cell types, at scales ranging from subcellular compartments, individual cells, local multi-cellular niches, to whole-brain regions from each slice. This work has two steps: 1) image registration to warp binary atlas based on landmarks provided manually. 2) image segmentation to leverage this characteristic of brain which different types of cells exhibit different image characteristics in different brain regions, and provide more accurate boundaries which leads to more robust landmarks to refine the registration results.

4.2 Rat Brain Atlas

The established and most commonly used atlases of rat brain are Paxinos and Watson [2], and Swanson [3] as shown in Fig. 4.1. These atlases provide series of sections, cut at specific angels, with external surface and internal boundaries of areas. Each section given a coordinate relative to the reference points on the skull (see Fig. 4.2 by George Paxinos[2]). Thee three most commonly used reference points in the rat are bregma, lambda, and the interaural line. Bregma is the anatomical reference point on the skull where the coronal suture is intersected perpendicularly by the sagittal suture. Lambda is the anatomical reference point on the skull where sagittal suture intersects the interaural line. Interaural line is the straight line between the points of ear bars in the external auditory meatus of each ear. Fig. 4.3 represents Waxholm

Space (WHS) on the atlasing template to provide standard spatial reference provided by [4]. As it is shown in the Fig. 4.3, brain atlases can illustrate coronal, sagittal or horizontal sections of the brain. The Paxinos and Watson atlas is based on a coronal set which includes 161 sections from a single brain at regular 0.12 mm intervals. Swanson atlas was built on the same stereotaxic coordinates of the Paxinos atlas and contains 73 coronal sections which are unequally placed and makes it challenging to use when there is a new slice of rat brain to analyze.

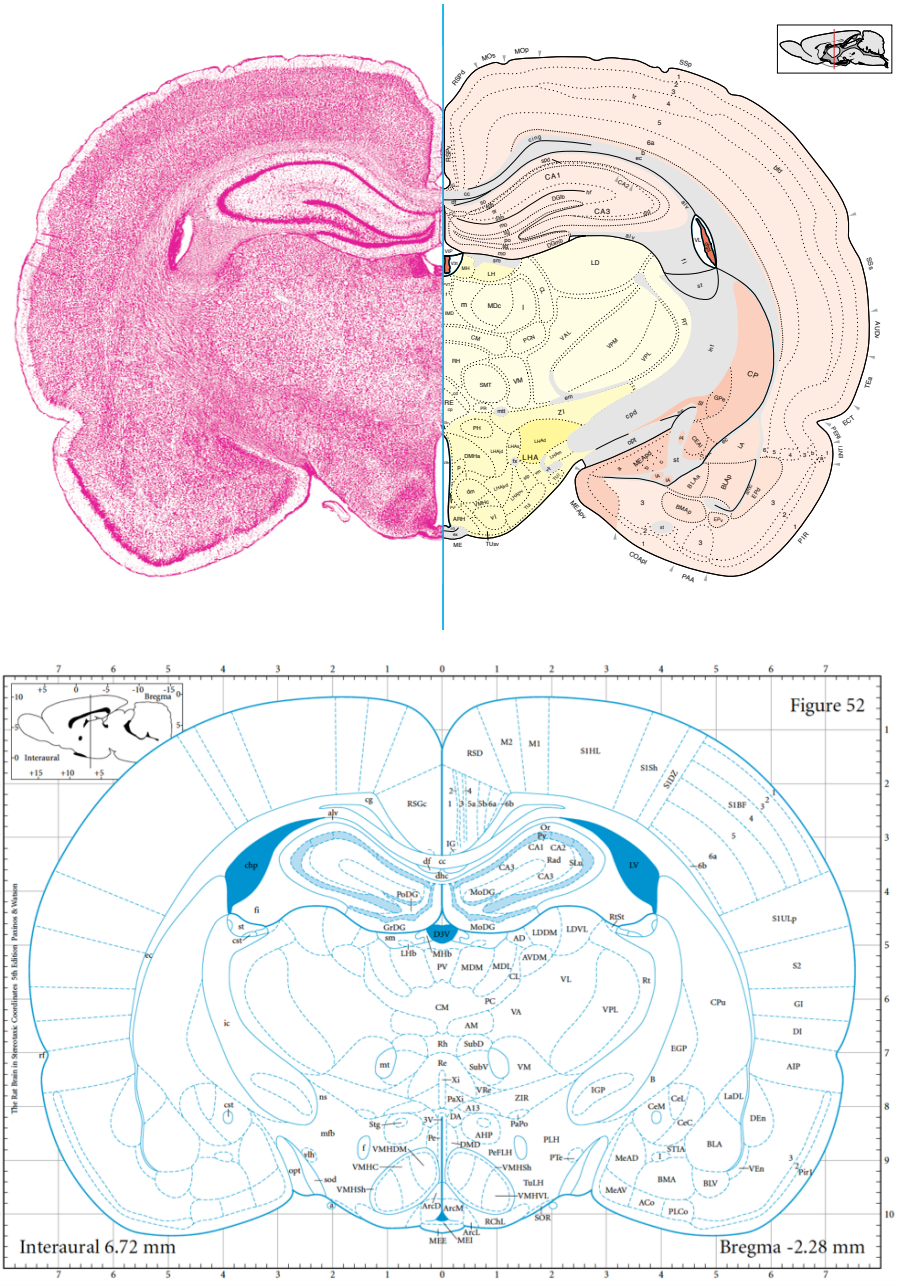


Figure 4.1: Swanson (top) and Paxinos (bottom) atlases for bregma values of 2.50mm ± 0.12 mm; Adopted from [2] and [3].

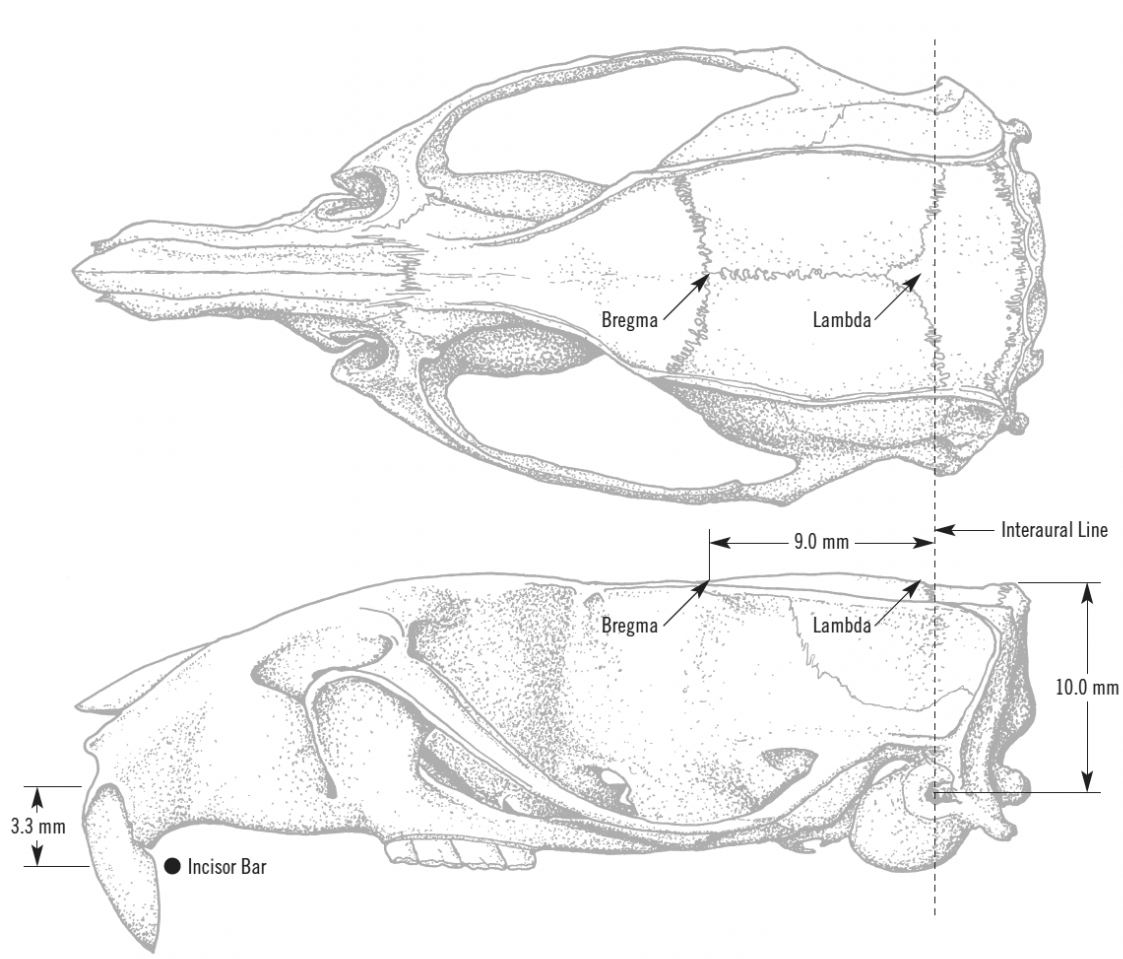


Figure 4.2: Dorsal (top) and lateral (bottom) views of the skull of a rat. The positions of bregma, lambda and the plane of the interaural line are shown above the lateral view. Adopted from [2].

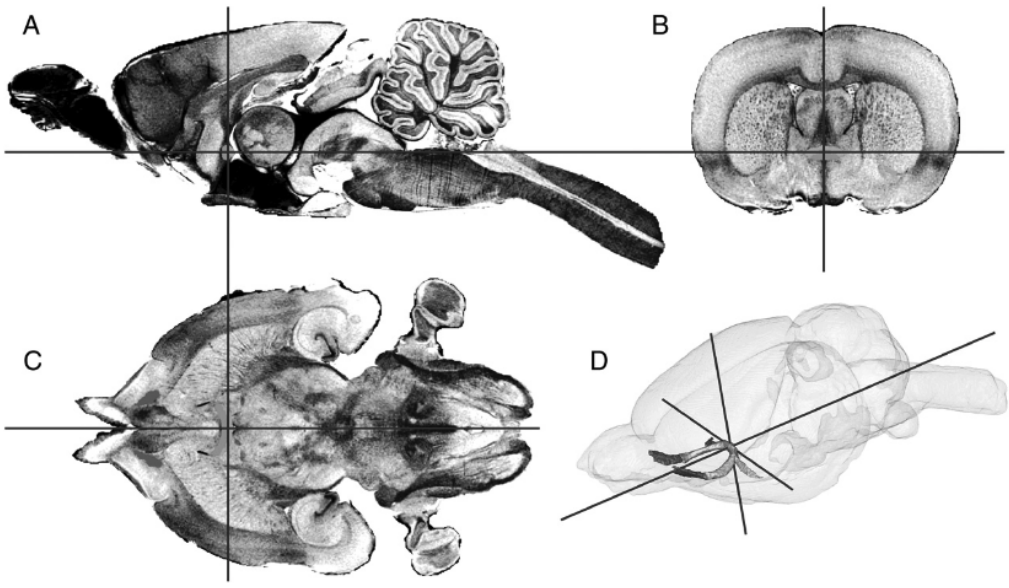


Figure 4.3: Waxholm Space atlas of the Sprague Dawley rat brain. sagittal view (A), coronal view (B),and horizontal view (C), overall position of the origin within the brain (D); Adopted from [4].

4.3 Dataset Description

A multiplexed imaging technique shown in Fig. 4.4 captures a comprehensive molecular signature for each cell. In this imaging technique, a high-detail imaging is utilized to illustrate changes in extended regions (whole coronal/sagittal brain slices).

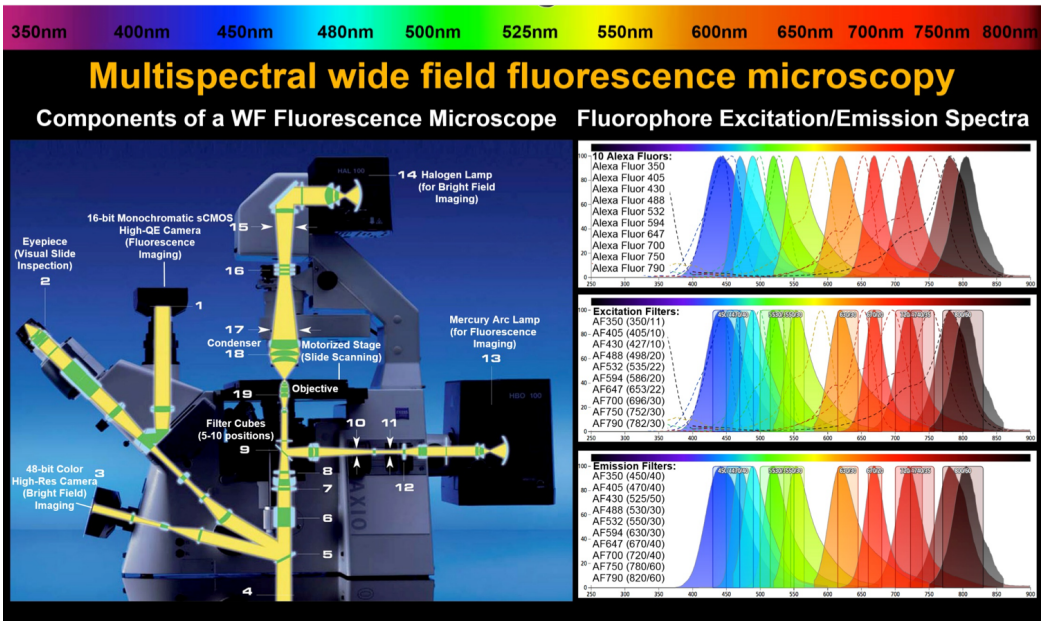


Figure 4.4: Multiplex Fluorescence Microscopy; Adopted from [5].

To obtain this dataset, iterative rounds of antibody staining and imaging are performed on the same tissue carefully.There are five rounds of 10plex immune-labeling images for each rat brain tissue as shown in Fig. 4.5. Combinations of these biomarkers can describe a large universe of cell types and cell states. The application of such techniques leads to the ability of analyzing each cell in the level of cell state, protein expression levels, alterations to protein sub-cellular localization, morphological alterations and etc.

A whole slice of brain has a size of $\sim$ 2.4 Gb with $\sim 32,000 \times 47,000$ pixels. Each cell in this image is $\sim 40 \times 40$ pixels for one channel in a 50 channel dataset. Because this images are massive datasets, working with the morphological and textural features of each channel in the image requires memory efficient and computationally fast techniques for data processing.

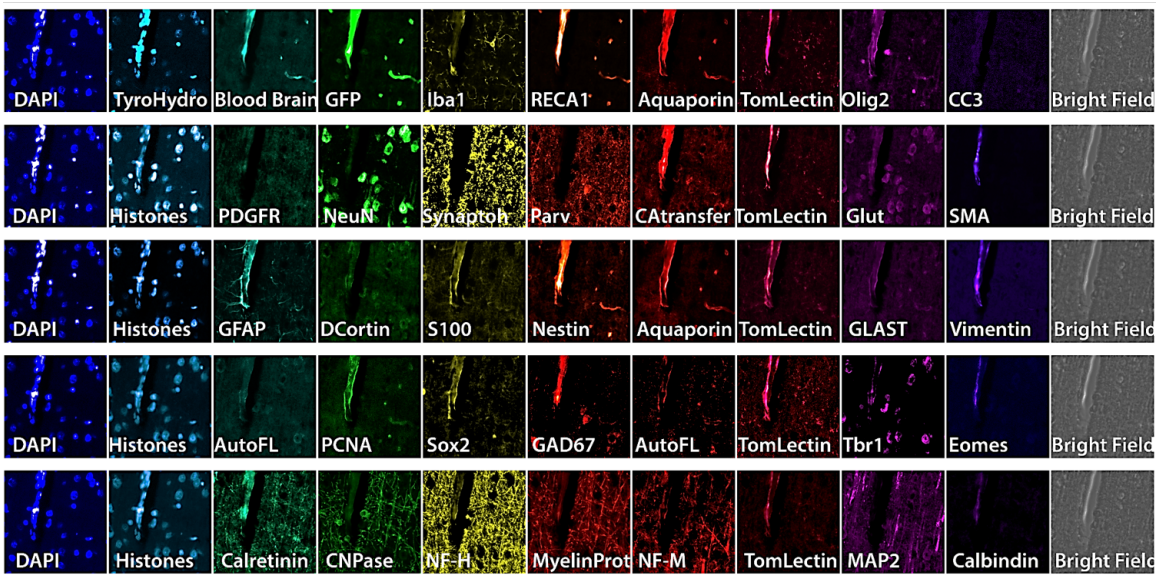


Figure 4.5: Multiple Rounds of Imaging; Adopted from [5].

4.4 The Proposed Deep Learning Framework for Brain Region Segmentation

Fig. 4.6 demonstrates the system flowchart of proposed work for fluorescence microscopic multiplex rat brain images. This work consists of three main steps: A) Training the segmentation model with 40×40 patches extracted from 10 selected regions (Rt, CC, MHB, mt, stg, igp, vmh, opt, fi, sm) of the training image. B) Initial localization of the defined regions in the test image with registration done by manually selected landmarks and dilation, and C) Apply trained U-Net to predict pixels in the selected regions of the test image.

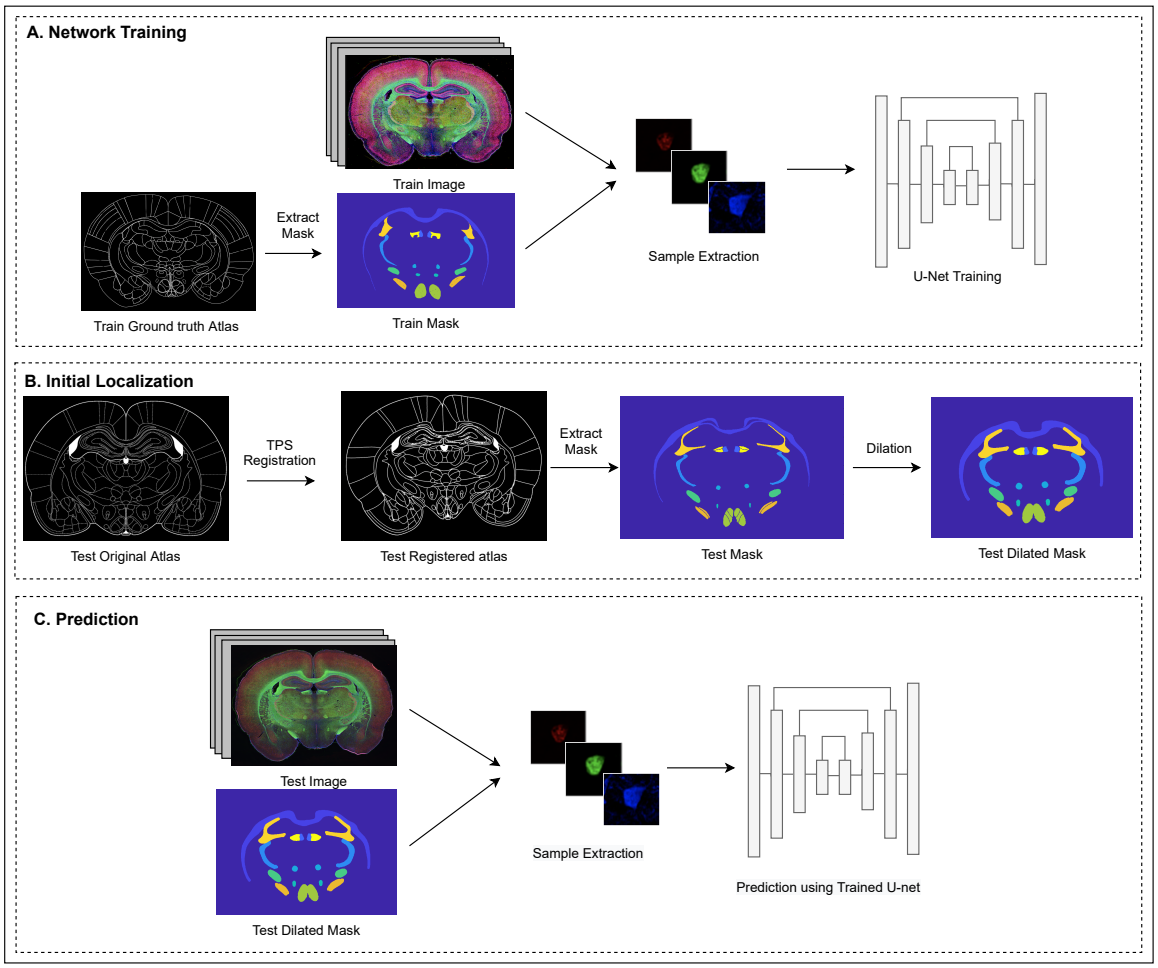


Figure 4.6: The Framework of Deep Learning method for brain region segmentation; A) training the segmentation model with samples extracted from selected regions of the train image, B) initial localization of the brain regions in the test image with registration done by manually selected landmarks and dilation, C) Apply trained U-Net to predict pixels in selected regions of the test image.

4.4.1 Deformable Image Registration

Deformable image registration involves estimating the geometric transformation between two images to map them onto a common coordinate system. The process is deformable, or nonlinear, because the estimated transformation does not include only rigid transformations such as translation and/or rotation but also deformations like shrinking and/or stretching. Deformable image registration can be categorized

into landmark-based and feature-based registration methods. Feature-based methods are based on boundary, surface and edges information of two images from same type of data. On the other hand, landmark-based methods are based on coordinate transformation (rotation, translation, and scaling) and interpolation computed from some control points called landmarks, and mostly used to register images with different type of data. In this work, we use a landmark-based registration method to warp the Paxinos binary atlas because the atlas and multiplexed rat brain image are not from same data type. Land-mark based deformable image registration can be based on global interpolation in which each movement in a landmark is broadcasted to all other landmarks (e.g., thin plate spline interpolation), while registration methods based on local interpolation only consider the adjacent landmarks to tract the computational overload (e.g., b-spline based registration).

Thin Plate Spline (TPS): There are many image registration algorithms based on matching corresponding landmarks (control points) in two images. TPS image registration technique[70] is the most commonly used landmark driven image registration algorithm. The use of thin-plate spline interpolation as a landmark-based deformable registration method of medical images was first introduced by Bookstein[70]. The TPS algorithm defines a unique smooth registration from a source image to a target image. TPS define a unique correspondence between the two images only at the landmark points. TPS is a spline interpolation method where the smoothness of the spline is controlled by a gradient based regularization term. Below we briefly describe this method in the general context of d -dimensional images. The problem can be stated as follows:

Given two sets of n landmarks $\mathbf{p}_i$ and $\mathbf{q}_i, i = 1, \ldots, n$ in two image of dimension d. Within a suitable Hilbert space H^d of allowable functions find the transformation $\mathbf{u}$, which a) minimizes a given functional $J : H^d \to \mathbb{R}$ and b) fulfills the interpolation conditions

$$\mathbf{u}\left(\mathbf{p}_i\right) = \mathbf{q}_i, \quad i = 1, \ldots, n. \tag{4.1}$$

The problem of finding the transformation $\mathbf{u}$ can be subdivided into d problems for each component z of $\mathbf{u}$. The functional J is fully described through the dimension d of the domain and the order m of derivatives used. We will write J_m^d for these functionals

$$J_m^d(z) = \sum_{\alpha_1+\ldots+\alpha_d=m} \frac{m!}{\alpha_1! \cdots \alpha_d!} \int_{\mathbb{R}^d} \left(\frac{\partial^m z}{\partial x_1^{\alpha_1} \cdots \partial x_d^{\alpha_d}} \right)^2 d\mathbf{x} \tag{4.2}$$

Let a set of functions ϕ_i span the space $\Pi^{m-1}\left(\mathbb{R}^d\right)$ of all polynomials on $\mathbb{R}^d$ up to order $m-1$, which is the nullspace of the functional J_m^d. The dimension of this space is

$$M = \left(\begin{array}{c} d+m-1 \\ d \end{array} \right) \tag{4.3}$$

and must be lower than n (the minimum number of landmarks). The solution of the minimization problem can now be written in the following form

$$z(\mathbf{x}) = \sum_{i=1}^{M} a_i \phi_i(\mathbf{x}) + \sum_{i=1}^{n} w_i U_i(\mathbf{x}) \tag{4.4}$$

with some basis functions $U_i = U\left(\cdot, \mathbf{p}_i\right)$ depending on a) the dimension d of the domain, b) the order m of the functional J to be minimized and c) the Hilbert-space H of admissible functions. If we choose the Sobolev-space $H = \mathcal{H}^2$, which consists of all square integrable functions, with derivatives up to second order in $L_2(\mathbb{R})$, we obtain the kernel

$$U(\mathbf{x}, \mathbf{p}) = \left\{ \begin{array}{ll} |\mathbf{x}-\mathbf{p}|^{2m-d} \ln |\mathbf{x}-\mathbf{p}| & d \text{ even} \\ |\mathbf{x}-\mathbf{p}|^{2m-d} & \text{otherwise.} \end{array} \right. \tag{4.5}$$

The basis functions U_i span an n-dimensional space of functions that depend only on the source landmarks. The coefficient vectors $\mathbf{a} = (a_1, \ldots, a_M)^T$ and $\mathbf{w} = (w_1, \ldots, w_n)^T$ are computed as follows

$$\begin{aligned} \mathbf{K}\mathbf{w} + \mathbf{P}\mathbf{a} &= \mathbf{v} \\ \mathbf{P}^T\mathbf{w} &= 0 \end{aligned} \tag{4.6}$$

where $\mathbf{v}$ is the column vector of one component of the coordinates of the target points $\mathbf{q}_i$, and

$$K_{ij} = U_i\left(\mathbf{p}_j\right), \quad P_{ij} = \phi_j\left(\mathbf{p}_i\right). \tag{4.7}$$

4.4.2 Separable U-Net Model

As it is shown in Fig. 4.7, we propose to use depthwise separable convolution layers instead of standard convolutional layers in the U-Net model[71]. Depthwise separable convolutions were proposed first as a part of the Xception architecture[72]. Depthwise separable convolution is a form of factorization which factorizes a standard convolution into a depthwise convolution and a pointwise convolution (1×1 convolution). A standard convolution layer works by applying a convolution kernel to all of the channels of the input image and takes a weighted sum of the input pixels covered by the kernel sliding across all input channels of the image. This means that for a standard convolution, no matter how many input channels are available, the output channel is one. However, in depthwise separable convolutions, features are only learned from the input channels so the output layer has the same number of channels as the input. This is known as depthwise convolution followed by a pointwise (1×1) convolution layer which computes the weighted sum of all output channels into a single output. The U-Net architecture is designed as an improvement of the FCN architecture specifically for the segmentation of medical images. U-Net is symmetrical

and the layers that combine information from the encoding and decoding paths do so by concatenating the feature maps. The featuremaps are copied from the encoder to the corresponding decoder. These connections help in the accurate reconstruction of the mask, by copying the low-level feature maps from the encoder to the decoder and concatenate them to the corresponding upsampled layer. Convolving with these concatenated featuremaps helps the model recover spatial information that was lost due to pooling operations within the encoder.

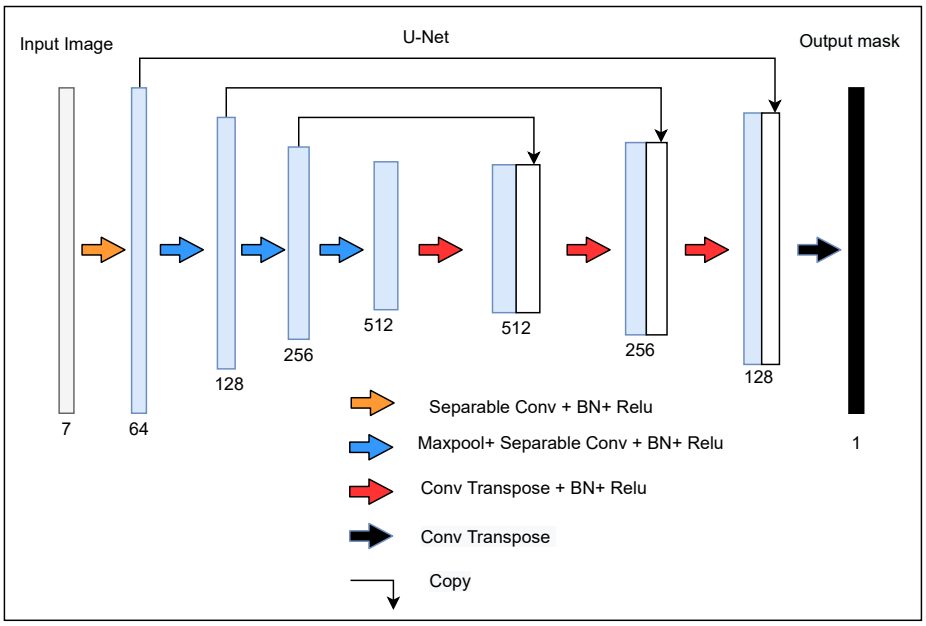


Figure 4.7: Separable U-Net with depthwise separable convolution layers; 7 channels: (NeuN, Parvalbumin, CNPase, Olig2, S100, GFAP, Tyrosine), number of filters in convolutional layers: 64, 128, 256, and 521.

4.4.3 Channel-specific U-Net Model

The idea of channel attention networks first introduced by Bastidas et al. [73], for semantic segmentation of Spacenet remote sensing multi-sensor dataset [74] to allocate attention away from noisy channels. There is another work by Chen et al. [75], introduced the concept of merging with an attention network to merge streams from multiple scales of the same image. In this work, an attention mechanism is to

learns to softly weight the channels at each pixel location of the image. To leverage the idea that each brain region of the multiplex rat brain image localize different biomarker, we design a channel attention U-Net shown in Fig. 4.9 to weight each channel (biomarkers) of the image during training the segmentation model and result in a better segmentation map.

Given an input image with several channels $b \in \{1, \ldots, B\}$. Each channel is passed through the U-Net and produces a score map for channel b, denoted as $f_{i,c}^b$ where i ranges over all the spatial positions and $c \in \{1, \ldots, C\}$ where C is the number of classes. We denote $y_{i,c}$ to be the weighted sum of score maps at spatial position i for class c for all the channels. The attention network in this work employs two convolutional layers, the first with a 5×5 kernel with N filters followed by a 1×1 Convolution with B filters and a pixel-wise Softmax

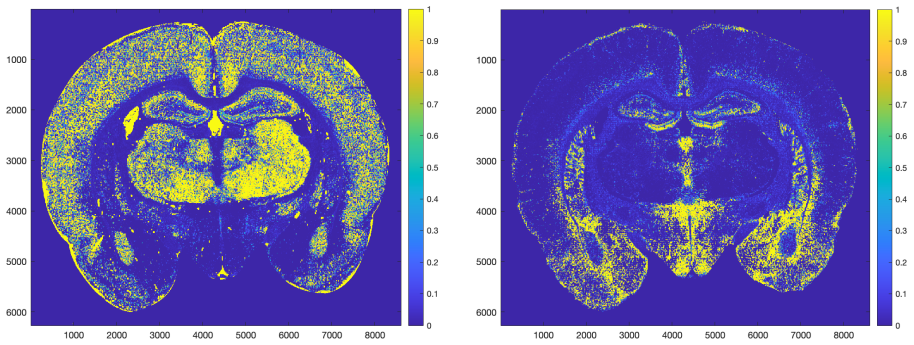
$$y_{i,c} = \sum_{b=1}^{B} w_i^b \cdot f_{i,c}^b \tag{4.8}$$

and the weight w_i^b is computed as follow

$$w_i^b = \frac{\exp\left(h_i^b\right)}{\sum_{t=1}^{B} \exp\left(h_i^t\right)} \tag{4.9}$$

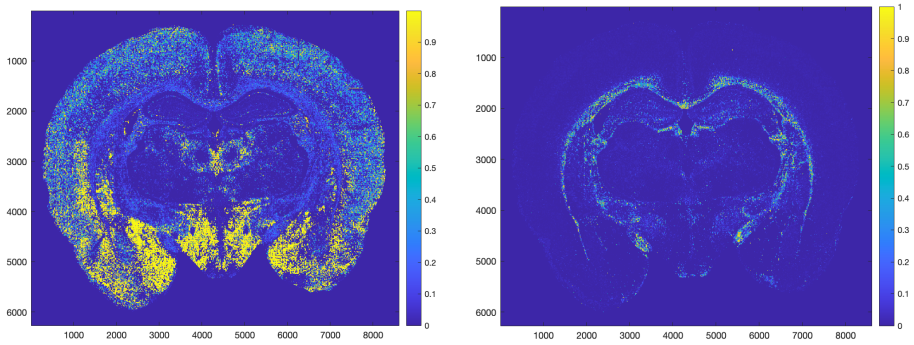
where h_i^b is the feature map produced by the attention network at position i for channel b.

The weight w_i^b reflects the importance of feature at channel b and position i. The attention model decides how much attention to pay to features at different spatial locations and channels. By visualizing w_i^b, we can visualize the attention for each channel as shown in Fig. 4.8.



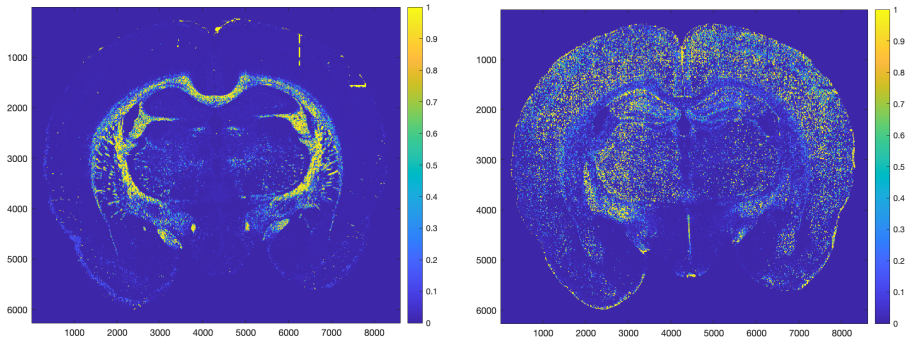
(a) Parvalbumin

(b) CNPase



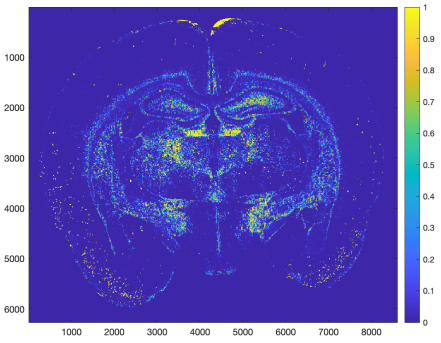
(c) GFAP

(d) NeuN



(e) Olig2

(f) Tyrosine



(g) S100

Figure 4.8: visualizing the attention weights for specified channels (Parvalbumin, CNPase, GFAP, NeuN, Olig2, Tyrosine, and S100) - The attention model decides how much attention to pay to features at different spatial locations and channels.

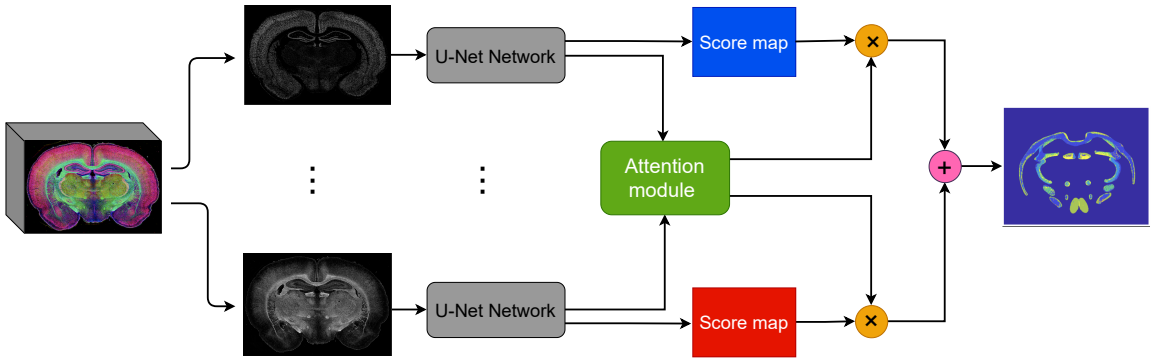


Figure 4.9: Proposed Channel-attention U-Net; Each channel is passed through the U-Net segmentation model and produce a score map fed into channel attention network to softly weight the channels at each pixel location of the image.

4.4.4 Decision Fusion Segmentation Model - Phonotype Features and Brain Image

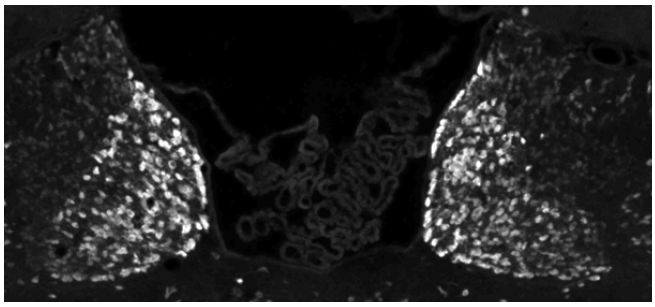
Based on spatial characteristic of the multiplex rat brain image , we found out that each region of the brain localize specific biomarkers with different cell properties. We have some experiments to see if we can leverage the phenotype based features from cell segmentation and cell detection of the rat brain image to improve our region segmentation method.

Cell density is the most common features which vary across different regions in the brain as shown in Fig. 4.10. We use the change in density for Neuron, Astrocyte, and Oligodendrocyte cells in different region of the brain as feature for semantic segmentation. We use the NeuN, S100, and Olig2 channels after registration and image correction. Cells are detected using multiplex classification [5]. We feed these density features to a U-net with fully connected layers. The predictions from this model (P_{FU}) are then fused with Separable-Unet model predictions (P_{SU}) as is shown in Fig. 4.11. In this method, after calculating the predictive probabilities for both networks: $P_C(y = k|\mathbf{x})$ and $P_G(y = k|\mathbf{x})$, the final predictive probability is computed

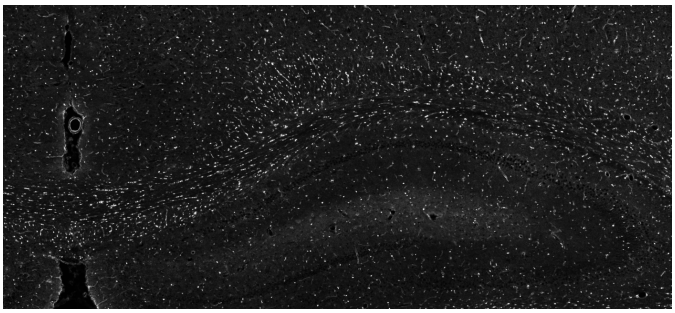
using decision fusion, such as Linear Opinion Pooling (LOP):

$$P(y = k|\mathbf{x}) = w_1 P_{SU}(y = k|\mathbf{x}) + w_2 P_{FU}(y = k|\mathbf{x}) \tag{4.10}$$

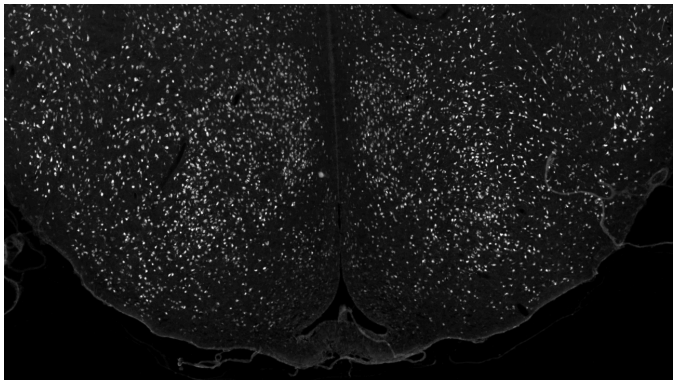
where w_1 and w_2 are positive weights which satisfy $w_1 + w_2 = 1$



Density of neuron in MHb region



Density of Oligodendrocyte in CC region



Density of neuron in VMH region

Figure 4.10: Cell density is the most common features which vary across different regions; Some examples that show change in cell density of some regions compared to their neighborhood area.

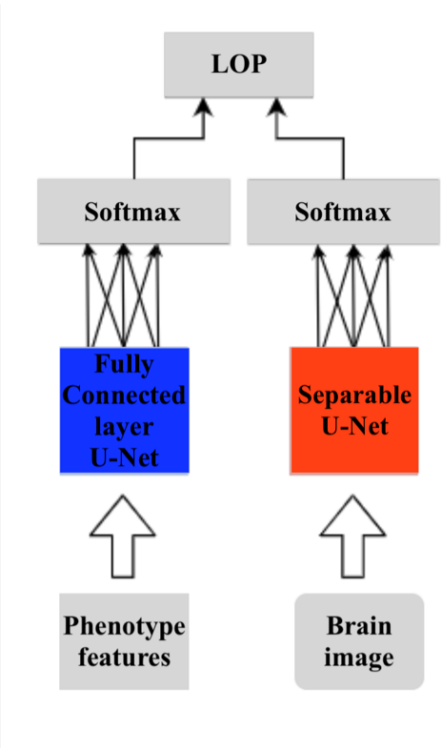


Figure 4.11: Decision fusion U-Net; The predictions from fully connected layer U-Net model (P_{FU}) are fused with Separable-Unet model predictions (P_{SU}).

4.4.5 Experimental Setup

Train and Test Sample and Label Extraction: In the field of neuroscience, the analysis of brain space and information commonly requires the segmentation of multiple brain regions which are distributed throughout the brain. Here, we selected ten brain regions from two Fluorescence microscopic 50plex images, which one was used for training and the other one was used for validation. Those ten selected brain regions (Rt, CC, MHb, mt, stg, IGP, VMH, opt, fi, sm) have visible differences in the surrounding areas. In this work, we only used seven channels (NeuN, Parvalbumin, CNPase, Olig2, S100, GFAP, Tyrosine) of 50plex images as shown in Fig. 4.12. For each brain region, 40 $\times$ 40 patches were extracted as the training and predicting images.

Table 4.1: Name of brain regions and their corresponding abbreviations based on Paxinos and Watson atlas

Abbreviation	Brain Region name
MHb	med habenular nu
mt	mammillothal tr
opt	optic tract
Rt	reticular th nu
sm	stria medullaris
CC	corpus callosum
IGP	int globus pallidus
fi	fimbria of hipp
Stg	stigmoid hy nu
VMH	ventromed nu

Initial Localization by Image Registration: It is necessary to locate brain regions before predicting the segmentation result to avoid over-segmentation and to improve efficiency. We first used Paxinos atlas to locate the brain region by mapping the corresponding labeled brain region space (atlas) to the new images. First, we registered the atlas to obtain the transformation. Then, the label for corresponding brain region from the atlas is extracted, and the label is mapped to the new image with the transformation, which enables general localization of brain regions. However, due to differences in biological samples and imaging mode, it can be difficult to guarantee an accurate match between the mapped label and brain region, especially where brain regions appear and disappear. So, we perform a dilation of the label to eliminate registration errors, which ensures that all pixels within the brain region are included in the dilated label. Fig. 4.13 represents the localization process of this work which consists of TPS registration, mask extraction and mask dilation. Fig. 4.14 shows the ground truth mask extracted from registered atlas of validation image and corresponding label.

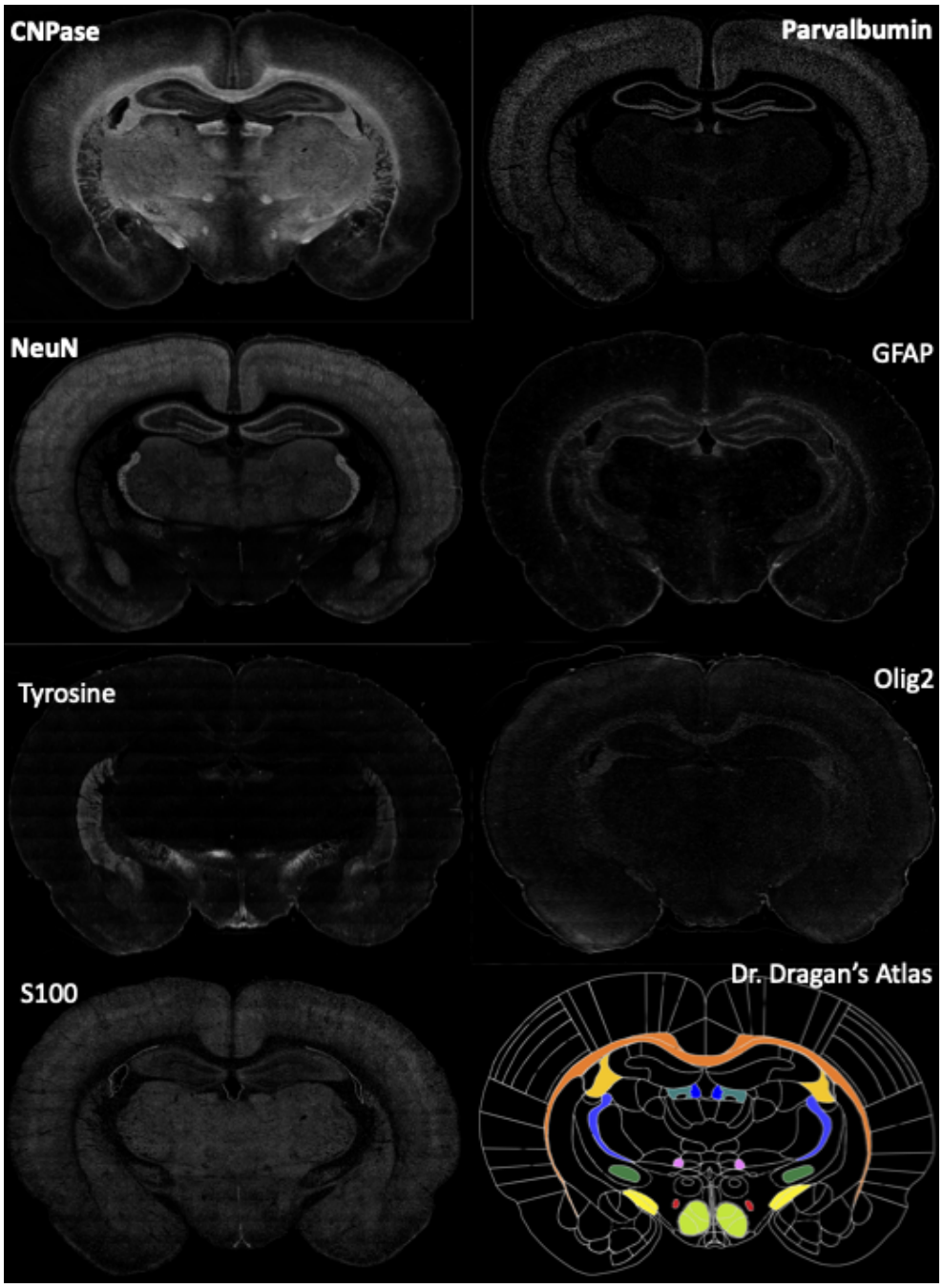


Figure 4.12: Selected regions (Rt, CC, MHb, mt, stg, IGP, VMH, opt, fi, sm) and channels (NeuN, Parvalbumin, CNPase, Olig2, S100, GFAP, Tyrosine) for region semantic segmentation; colored regions in the Dr. Dragan's atlas specify the selected regions.

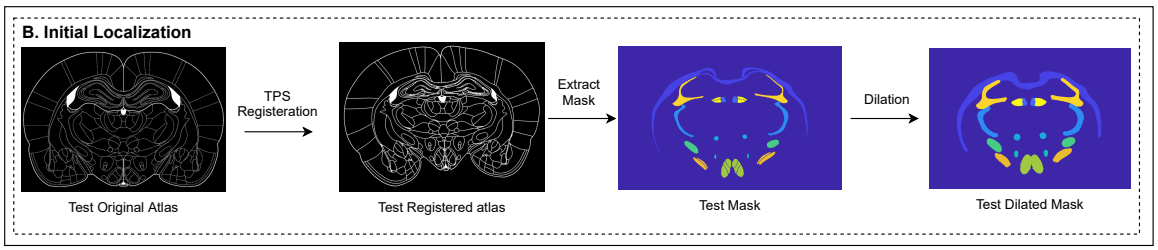


Figure 4.13: Initial Localization done by Image Registration and Dilation; After the test atlas warped using manually selected landmarks and TPS registration method, selected regions are extracted from the atlas and dilated to cover all the neighborhood around each region.

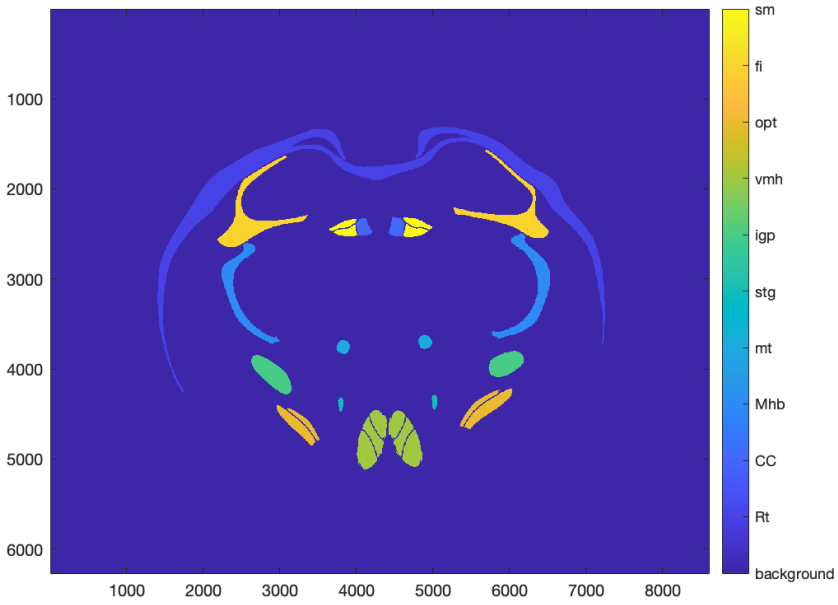


Figure 4.14: Ground truth extracted from registered atlas of validation image.

4.4.6 Performance Evaluation of Segmentation Methods

After extracting train patches and labels from one 50plex images and training the channel-wise U-Net, we used the trained model to predict another 50plex image. Although there are differences in the characteristics among each brain region, channel-wise U-Net displays good segmentation effects on most of these regions which can be shown in Fig. 4.15. The segmented lines are close to the real boundaries in the detail

images which this can help us to relocate the initial landmarks and define new set of landmarks based on the segmented lines to improve the registration accuracy.

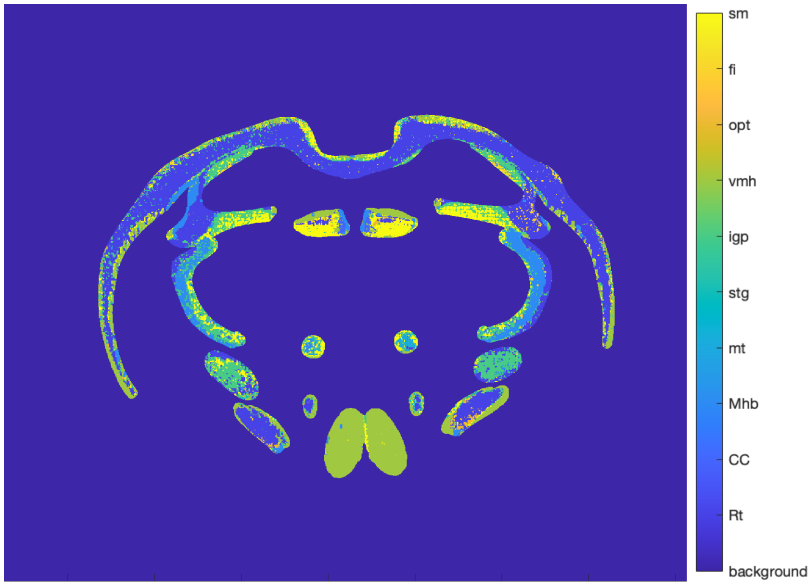


Figure 4.15: Segmentation result based on validation dilated mask.

However, the segmentation mask is noisy due to some miss segmentation which is need to be clean before detecting the segmented lines. As it is shown in Fig. 4.16, we use a median filter [76] to denoise the segmentation mask. Median Filtering is a nonlinear process useful in reducing salt and pepper noise. Median Filter is preserving edges in an image during the random noise reduction. In a median filter, a window with size of (i, j) slides along the image I, and the median intensity value of the pixels within the window becomes the output intensity $(I'(u, v))$ of the pixel $(I(u, v))$ being processed

$$I'(u, v) \leftarrow \text{median}\{I(u + i, v + j) \mid (i, j) \in R\}. \tag{4.11}$$

The median is calculated by first sorting all the pixel values from the surrounding neighborhood into numerical order and then replacing the pixel being considered with the middle pixel value. If the neighborhood under consideration contains an

even number of pixels, the average of the two middle pixel values is used.

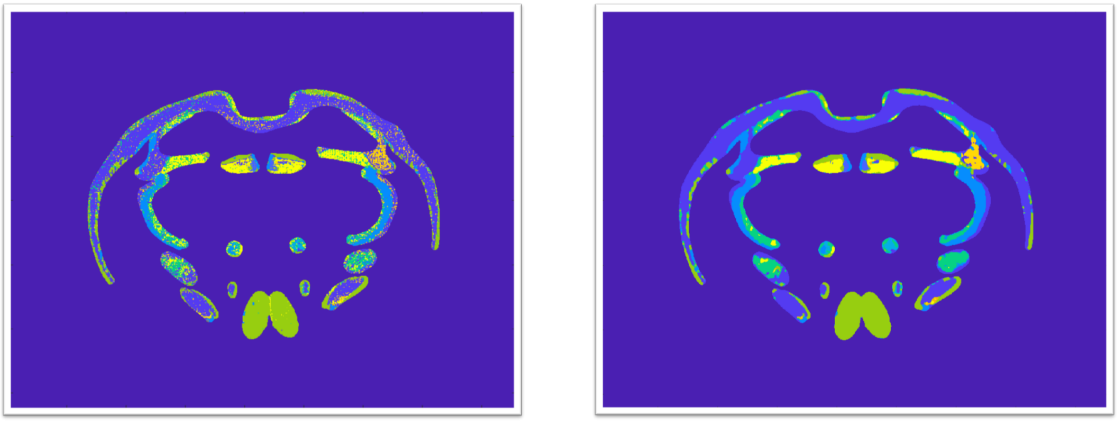


Figure 4.16: (left) noisy segmentation mask, (right) segmentation mask after median filter denoising.

After denoising the segmentation results, we apply a canny edge detection [77] to the image to find edges and dilate the results for a better visualization of segmented boundaries. Canny edge detection specified several criteria for the performance of edge detectors: 1) Minimum number of false negatives and false positives, 2) single response to single edge, and 3) report edge location at correct position. The result is shown in Fig. 4.17.

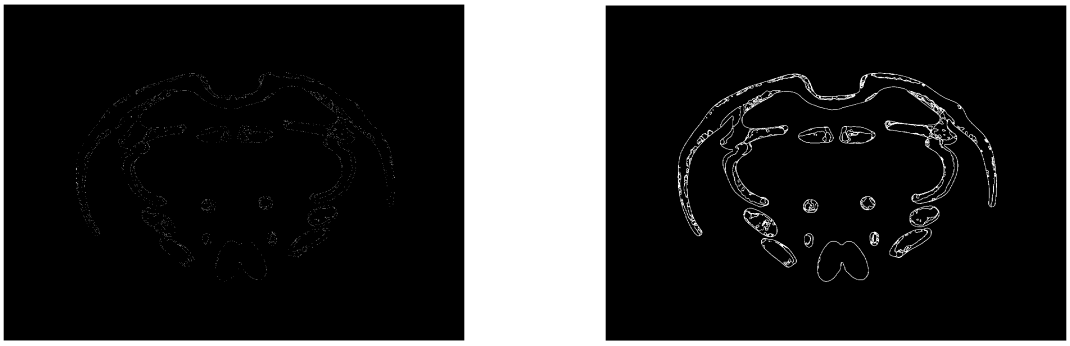


Figure 4.17: (left) Detected edges from segmentation mask, (right) Detected edges from segmentation after dilation.

After extracting the segmented lines, we use them as a new reference line to relocate the existing registration landmarks and define a new set of landmarks for those ten segmented regions. Fig. 4.18 and Fig. 4.19 shows the initial and new set of registration landmarks and their corresponding computed registered atlases. If we compare the new registered atlas and initial registered atlas, we can find significant improvements in atlas alignment. So we can say that multiplex rat brain region segmentation helps us to improve the registration process of the atlas. It enables biologically meaningful interpretations from computationally-derived results.

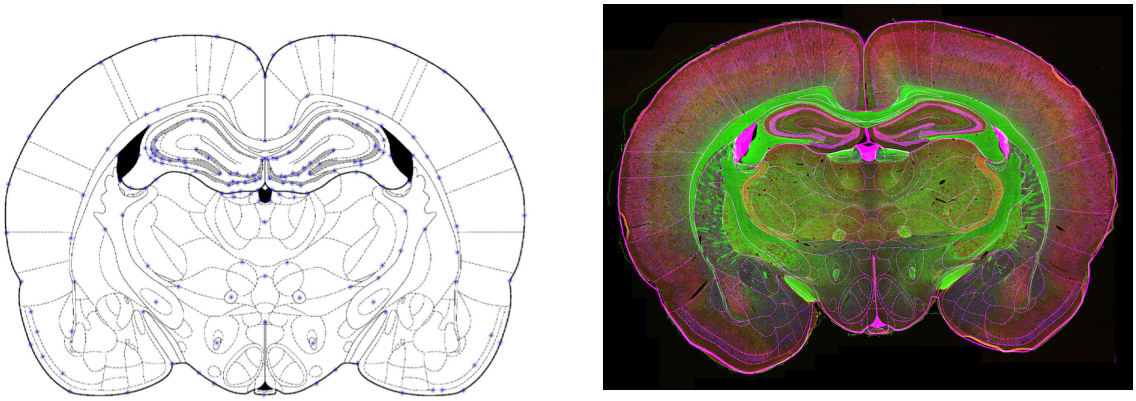


Figure 4.18: Initial registration landmark set and registered atlas overlaid with multiplex rat brain image - (left) Paxinos atlas overlaid with selected landmarks, (right) Registered atlas overlaid with corresponding rat image.

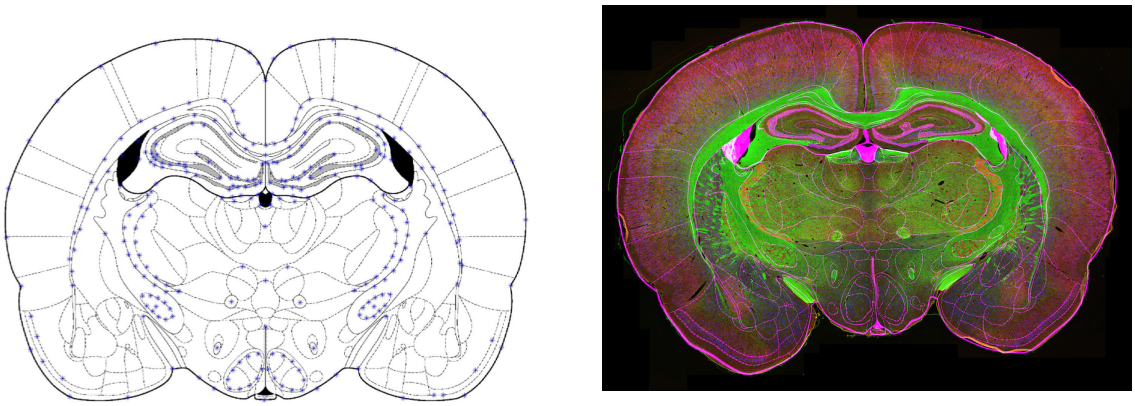


Figure 4.19: New registration landmark set and new registered atlas overlaid with multiplex rat brain image (left) Paxinos atlas overlaid with selected landmarks, (right) Registered atlas overlaid with corresponding rat image.

Segmentation Performance Measurements: In this work, the contour accuracy contributes to the perceived segmentation quality, and cannot captured by pixel based measurements. Therefor, we use a novel semantic contour-based accuracy inspired by standard contour metrics proposed in [78]. They focus on the $F1$-measure, precision and recall and extend Berkeley contour matching score [79] to semantic segmentation: they make it class-dependent by computing one value per class between the corresponding binarized segmentation maps. P^c and R^c represent the the precision and recall for each class, respectively

$$P^c = \frac{1}{\left|B^c_{ps}\right|} \sum_{z \in B^c_{ps}} \left[\left[d\left(z, B^c_{gt}\right) < \theta\right]\right] \tag{4.12}$$

and

$$R^c = \frac{1}{\left|B^c_{gt}\right|} \sum_{z \in B^c_{gt}} \left[\left[d\left(z, B^c_{ps}\right) < \theta\right]\right] \tag{4.13}$$

where $d(.)$ is the Euclidean distance, z represents pixel of image, B^c_{gt} is the boundary map of the binarized ground truth segmentation map for class c, and B^c_{ps} is the contour map for the binarized predicted segmentation map. θ is the distance error

tolerance. [[.]] is the Iversons bracket notation [80], where $[[b]] = 1$ if b=true and 0 otherwise. The $F1$-measure for class c is defined as follows

$$BF = F_1^c = \frac{2 \cdot P^c \cdot R^c}{R^c + P^c}. \tag{4.14}$$

And Finally to obtain per image $F1$ score (BF), F_1^c scores over all the classes either in the ground truth map or in the predicted segmentation map are averaged and provide a single value per image.

This method has two main drawbacks. First, it disregards the content of the segmentation beyond the tolerance threshold θ under which boundaries are matched. Second,the results of this metric depends on a discrete filtering of the distribution of boundary distances, so that the same score is obtained for different segmentation with different perceptual quality as far as the same amount of boundary pixels are within the threshold θ.

Fig. 4.20, Fig. 4.21, and Fig. 4.22 show the segmentation boundary overlaid on the brain image for Separable U-Net, Channel Attention U-Net, and Decision Fusion U-Net, respectively. If we compare the segmentation boundaries for all the proposed segmentation models, we can see the compared to Separable U-Net, Channel attention U-net and Decision fusion U-Net provided more accurate boundaries for some of the brain regions. The Contour based precision, recall, and F1 score for evaluation of segmentation boundaries from various segmentation models are shown in Table 4.2.

Table 4.2: Contour based precision, recall, and F1 score for evaluation of segmentation boundaries from various segmentation models

Segmentation model	Contour-based precision	Contour-based recall	Contour-based score
U-Net	0.95 (±0.01)	0.95 (±0.009)	0.95 (±0.009)
Separable U-Net	0.98 (±0.007)	0.91 (±0.009)	0.94 (±0.008)
Channel-Attention U-Net	0.98 (±0.009)	0.92 (±0.006)	0.95 (±0.007)
Decision Fusion	0.99 (±0.004)	0.91 (±0.003)	0.95 (±0.004)

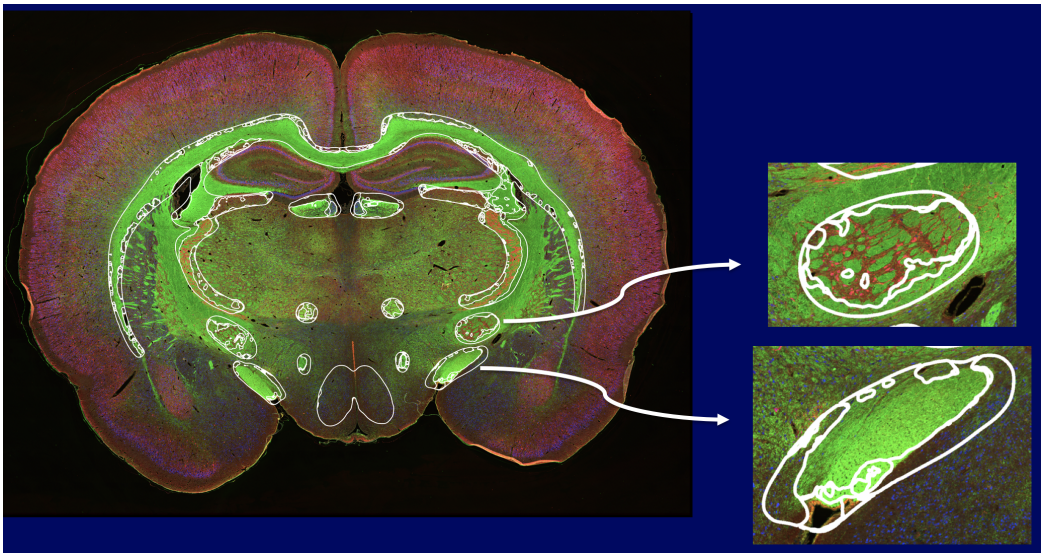


Figure 4.20: Separable U-Net segmentation contour result overlaid on the brain image.

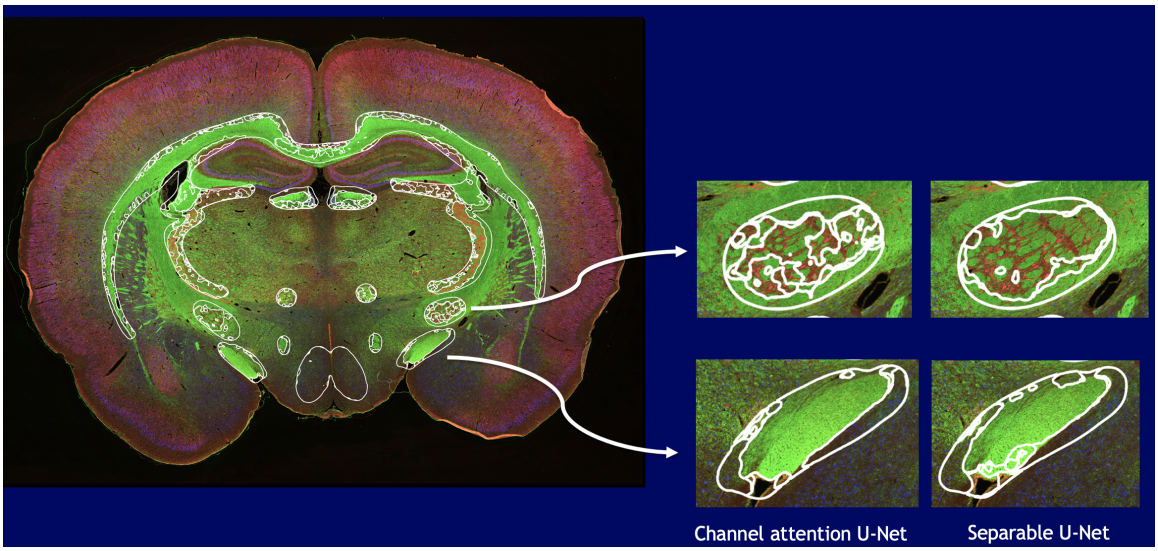


Figure 4.21: Channel-attention U-Net segmentation contour result overlaid on the brain image.

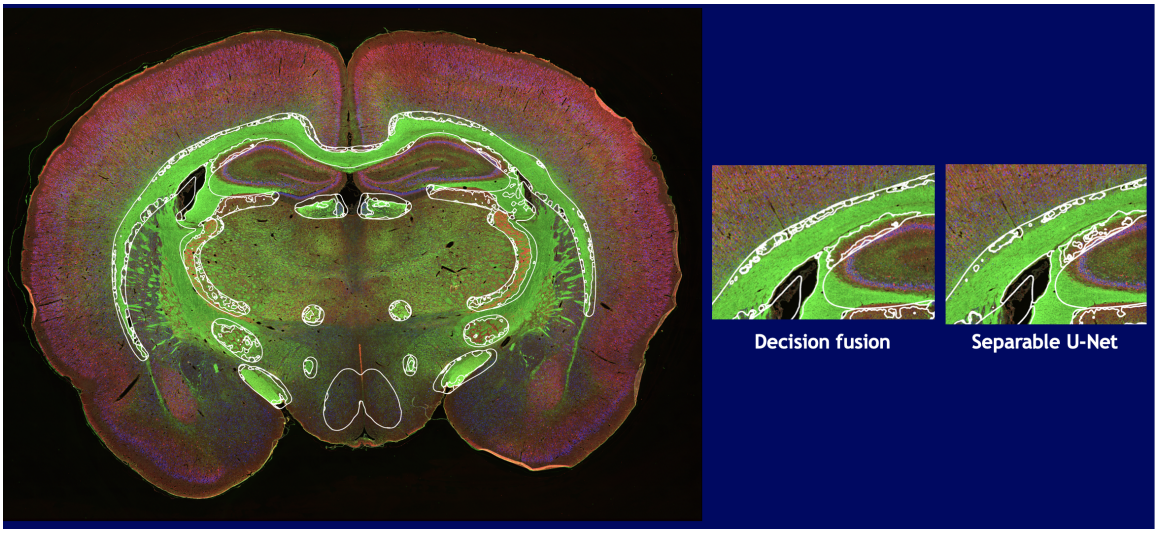


Figure 4.22: Decision fusion U-Net segmentation contour result overlaid on the brain image.

4.5 Cellular Profiling of Brain Regions

We perform quantitative profiling of the brain regions with respect to their cell composition and cell phenotypic status. Table 4.5 shows profiling of neurons (and subtypes), astrocytes, microglia, oligodendrocytes and endothelials for Rt, CC, MHb, mt, stg, IGP, VMH, opt, fi, sm regions of healthy 50plex rat brain. Cellular profiling can be used to quantify and analysis treatment experiments specific to the brain regions. Based on the cellular profiling of brain regions, we can see high density of neuronal cells in Mhb, VMH, and sm regions, and high density of oligodendrocyte cells in opt, fi , and cc regions.

Table 4.3: Cellular profiling of neurons, astrocytes, microglia, oligodendrocytes and endothelials for Rt, CC, MHb, mt, stg, IGP, VMH, opt, fi, sm regions of healthy 50plex rat brain. All cellular densities are reported per mm^2.

Regions	Neuron	Astrocyte	Microglia	Endothelial	Oligodendrocyte
CC	177.116567	354.8673921	382.9333119	275.902262	657.4084932
MHb	2157.794544	284.0514758	194.3510097	353.8185049	204.3177282
Rt	463.5892056	540.1093516	253.5777101	708.7887975	201.0748362
mt	479.6843197	289.9190944	195.0364817	347.9029132	189.7652254
stg	530.0528877	361.987338	206.8499074	426.627934	297.3467419
IGP	477.1298968	371.3651127	254.3108292	491.9845013	190.1389377
VMH	1065.197878	428.0407126	164.2807645	329.6123654	141.8629203
opt	258.0149549	274.1408896	403.148367	241.8890202	1032.05982
fi	39.27493839	243.504618	270.9970749	290.6345441	1386.405325
sm	761.1211287	483.2234143	168.154668	473.488144	245.1518054

4.6 Steps of Atlas Fitting Pipeline

The following steps summarize the atlas fitting pipeline.

- We select a 50 plex image S1 (Bregma of -2.28 mm) as the training image (more images can be incorporated into training if available). The ground truth atlas of this image was available (as annotations from subject-matter experts). We use Fiji (Bigwarp plugin) [81] for initial alignment (warping) of the atlas for the test image (the 50 plex image S2 with a Bregma of -2.64 mm).
- We define and annotate the training and test masks based on regions of interest.
- We extract training samples (40×40 patches) from the train-image based on the training mask computed from the ground-truth atlas.
- We extract validation samples (40×40 patches) from the test image based on the test mask computed from the initial aligned atlas.
- We train the segmentation model with training samples and run the trained model on the test image to segment the regions.

- We perform post-processing (denoising, edge detection, and edge dilation) after running trained segmentation model on test image.
- We overlay segmentation boundaries on the brain image and relocate the existing landmarks to lie on segmented boundaries. Additional landmarks can be added at this step based on well-defined segmentation boundaries.
- We undertake TPS registration to align the atlas using the new landmarks.

Chapter 5

Conclusion and Future Work

In this dissertation, we have developed multiple deep learning pipelines to analyze multi-channel images from remote sensing and FTIR biomedical hyperspectral images to multiplex rat brain images. Listed below is a summary of key contributions.

- **Convolutional neural networks for hyperspectral image analysis:** We focused on optimizing CNNs for practical applications involving both remote sensing and biomedical HSI. We investigate how to select appropriate network configurations and parameters for various type of HSI data. We also define a strategy to appropriately acquire training data of HSI remotely sensed images for an unbiased and accurate result. A variety of CNNs, RNNs and joint models are studied in this work. The information provided in this part of the dissertation can be used as a guide for future research in deep learning models for remote sensing and biomedical hyperspectral image analysis.

- **Graph based convolutional neural networks for hyperspectral image analysis:** We proposed a graph-based CNN architecture for classification of hyperspectral data in remote sensing and biomedical hyperspectral images. We propose a semi-supervised strategy to construct a robust adjacency matrix that is used by the graph convolutional networks - our approach leverages information provided by labeled training spectra and unlabeled spectra. We also developed a spatial-spectral fusion network where spatial convolutions are utilized in conjunction with graph convolutions along the spectral reflectance direction to learn both object-specific spatial features and the graph-structure of the spec-

tral features simultaneously. The proposed method exhibits higher classification performance compared to traditional CNNs and is promising for remote sensing applications with high spectral and spatial resolutions.

- **Atlas fitting** We combine image registration and segmentation methodologies to build a semi-automatic atlas fitting for rat brain image parsing. First we perform image registration to warp a binary atlas based on some landmarks selected manually. Following this, we utilize an image segmentation approach to leverage the fact that different regions of the brain exhibit different characteristics in the multiplex images – this provides more accurate boundaries which allows us to update our landmarks and refine the registration results recursively.

In future work, these experiments can be expanded to other datasets and imagery from healthy and unhealthy rat brains. More regions of the brain also need to be examined for creating a more comprehensive segmentation.

REFERENCES

[1] F. F. Shahraki and S. Prasad, "Graph convolutional neural networks for hyperspectral data classification," in *2018 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, Nov 2018, pp. 968–972.

[2] G. Paxinos and C. Watson, *The rat brain in stereotaxic coordinates: hard cover edition*. Elsevier, 2006.

[3] L. W. Swanson, "Brain maps 4.0—structure of the rat brain: An open access atlas with global nervous system nomenclature ontology and flatmaps," *Journal of Comparative Neurology*, vol. 526, no. 6, pp. 935–943, 2018.

[4] E. A. Papp, T. B. Leergaard, E. Calabrese, G. A. Johnson, and J. G. Bjaalie, "Waxholm space atlas of the sprague dawley rat brain," *Neuroimage*, vol. 97, pp. 374–386, 2014.

[5] D. Maric, J. Jahanipour, X. R. Li, A. Singh, A. Mobiny, H. Van Nguyen, A. Sedlock, K. Grama, and B. Roysam, "Whole-brain tissue mapping toolkit using large-scale highly multiplexed immunofluorescence imaging and deep neural networks," *Nature communications*, vol. 12, no. 1, pp. 1–12, 2021.

[6] Y. Ozaki and S. Kawata, *Far-and Deep-ultraviolet spectroscopy*. Springer, 2015.

[7] L. Gao and R. T. Smith, "Optical hyperspectral imaging in microscopy and spectroscopy–a review of data acquisition," *Journal of biophotonics*, vol. 8, no. 6, pp. 441–456, 2015.

[8] F. D. Van der Meer, H. M. Van der Werff, F. J. Van Ruitenbeek, C. A. Hecker, W. H. Bakker, M. F. Noomen, M. Van Der Meijde, E. J. M. Carranza, J. B. De Smeth, and T. Woldai, "Multi-and hyperspectral geologic remote sensing: A review," *International Journal of Applied Earth Observation and Geoinformation*, vol. 14, no. 1, pp. 112–128, 2012.

[9] S. Pahlow, K. Weber, J. Popp, R. W. Bayden, K. Kochan, A. Rüther, D. Perez-Guaita, P. Heraud, N. Stone, and A. Dudgeon, "Application of vibrational spectroscopy and imaging to point-of-care medicine: A review," *Applied spectroscopy*, vol. 72, no. 101, pp. 52–84, 2018.

[10] E. R. A. van Hove, D. F. Smith, and R. M. Heeren, "A concise review of mass spectrometry imaging," *Journal of chromatography A*, vol. 1217, no. 25, pp. 3946–3954, 2010.

[11] D. B. Le, "Molecular diffusion nuclear magnetic resonance imaging." *Magnetic resonance quarterly*, vol. 7, no. 1, pp. 1–30, 1991.

[12] J. B. Hearnshaw, *The analysis of starlight: one hundred and fifty years of astronomical spectroscopy.* CUP Archive, 1990.

[13] H. Huang, H. Yu, H. Xu, and Y. Ying, "Near infrared spectroscopy for on/in-line monitoring of quality in foods and beverages: A review," *Journal of food engineering*, vol. 87, no. 3, pp. 303–313, 2008.

[14] D. Yang and Y. Ying, "Applications of raman spectroscopy in agricultural products and food analysis: A review," *Applied Spectroscopy Reviews*, vol. 46, no. 7, pp. 539–560, 2011.

[15] Y. Roggo, P. Chalus, L. Maurer, C. Lema-Martinez, A. Edmond, and N. Jent, "A review of near infrared spectroscopy and chemometrics in pharmaceutical

technologies," *Journal of pharmaceutical and biomedical analysis*, vol. 44, no. 3, pp. 683–700, 2007.

[16] S. Berisha, F. F. Shahraki, D. Mayerich, and S. Prasad, "Deep learning for hyperspectral image analysis, part i: Theory and algorithms," *Hyperspectral Image Analysis*, p. 37, 2020.

[17] F. F. Shahraki, L. Saadatifard, S. Berisha, M. Lotfollahi, D. Mayerich, and S. Prasad, "Deep learning for hyperspectral image analysis, part ii: Applications to remote," *Hyperspectral Image Analysis: Advances in Machine Learning and Signal Processing*, p. 69, 2020.

[18] F. F. Shahraki and S. Prasad, "Joint spatial and graph convolutional neural networks - a hybrid model for spatial-spectral geospatial image analysis," in *IGARSS 2020 - 2020 IEEE International Geoscience and Remote Sensing Symposium*, 2020, pp. 4003–4006.

[19] W. Hu, Y. Huang, L. Wei, F. Zhang, and H. Li, "Deep convolutional neural networks for hyperspectral image classification," *Journal of Sensors*, vol. 2015, 2015.

[20] W. Zhao and S. Du, "Spectral–spatial feature extraction for hyperspectral image classification: A dimension reduction and deep learning approach," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 54, no. 8, pp. 4544–4554, 2016.

[21] J. Yue, W. Zhao, S. Mao, and H. Liu, "Spectral–spatial classification of hyperspectral images using deep convolutional neural networks," *Remote Sensing Letters*, vol. 6, no. 6, pp. 468–477, 2015.

[22] F. Zhang, B. Du, and L. Zhang, "Scene classification via a gradient boosting random convolutional network framework," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 54, no. 3, pp. 1793–1802, March 2016.

[23] M. Vakalopoulou, K. Karantzalos, N. Komodakis, and N. Paragios, "Building detection in very high resolution multispectral data with deep learning features," in *2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, July 2015, pp. 1873–1876.

[24] L. Zhang, Z. Shi, and J. Wu, "A hierarchical oil tank detector with deep surrounding features for high-resolution optical satellite imagery," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 8, no. 10, pp. 4895–4909, Oct 2015.

[25] W. Li, G. Wu, and Q. Du, "Transferred deep learning for anomaly detection in hyperspectral imagery," *IEEE Geoscience and Remote Sensing Letters*, vol. 14, no. 5, pp. 597–601, May 2017.

[26] C. Li, S. X. Yang, Y. Yang, H. Gao, J. Zhao, X. Qu, Y. Wang, D. Yao, and J. Gao, "Hyperspectral remote sensing image classification based on maximum overlap pooling convolutional neural network," *Sensors*, vol. 18, no. 10, 2018. [Online]. Available: http://www.mdpi.com/1424-8220/18/10/3587

[27] K. Makantasis, K. Karantzalos, A. Doulamis, and N. Doulamis, "Deep supervised learning for hyperspectral data classification through convolutional neural networks," in *Geoscience and Remote Sensing Symposium (IGARSS), 2015 IEEE International*. IEEE, 2015, pp. 4959–4962.

[28] Y. Li, H. Zhang, and Q. Shen, "Spectral–spatial classification of hyperspectral imagery with 3d convolutional neural network," *Remote Sensing*, vol. 9, no. 1, p. 67, 2017.

[29] Z. Zuo, B. Shuai, G. Wang, X. Liu, X. Wang, B. Wang, and Y. Chen, "Convolutional recurrent neural networks: Learning spatial dependencies for image representation," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2015, pp. 18–26.

[30] Y. Xiao and K. Cho, "Efficient character-level document classification by combining convolution and recurrent layers," *arXiv preprint arXiv:1602.00367*, 2016.

[31] P. J. Werbos, "Backpropagation through time: what it does and how to do it," *Proceedings of the IEEE*, vol. 78, no. 10, pp. 1550–1560, 1990.

[32] H. Wu and S. Prasad, "Convolutional recurrent neural networks forhyperspectral data classification," *Remote Sensing*, vol. 9, no. 3, p. 298, 2017.

[33] X. Yu, X. Wu, C. Luo, and P. Ren, "Deep learning in remote sensing scene classification: a data augmentation enhanced convolutional neural network framework," *GIScience & Remote Sensing*, vol. 54, no. 5, pp. 741–758, 2017.

[34] H. Wu and S. Prasad, "Semi-supervised deep learning using pseudo labels for hyperspectral image classification," *IEEE Transactions on Image Processing*, vol. 27, no. 3, pp. 1259–1270, March 2018.

[35] X. Zhou and S. Prasad, "Domain adaptation for robust classification of disparate hyperspectral images," *IEEE Transactions on Computational Imaging*, vol. 3, no. 4, pp. 822–836, Dec 2017.

[36] Y. Chen, Z. Lin, X. Zhao, G. Wang, and Y. Gu, "Deep learning-based classification of hyperspectral data," *IEEE Journal of Selected topics in applied earth observations and remote sensing*, vol. 7, no. 6, pp. 2094–2107, 2014.

[37] B. Fang, Y. Li, H. Zhang, and J. C.-W. Chan, "Hyperspectral images classification based on dense convolutional networks with spectral-wise attention mechanism," *Remote Sensing*, vol. 11, no. 2, p. 159, 2019.

[38] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," *arXiv preprint arXiv:1511.07122*, 2015.

[39] L. Mou, P. Ghamisi, and X. X. Zhu, "Deep recurrent neural networks for hyperspectral image classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 7, pp. 3639–3655, 2017.

[40] R. Hang, Q. Liu, D. Hong, and P. Ghamisi, "Cascaded recurrent neural networks for hyperspectral image classification," *IEEE Transactions on Geoscience and Remote Sensing*, 2019.

[41] C. M. Bachmann, T. L. Ainsworth, and R. A. Fusina, "Exploiting manifold geometry in hyperspectral imagery," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 43, no. 3, pp. 441–454, March 2005.

[42] R. Kimmel and J. A. Sethian, "Computing geodesic paths on manifolds," *Proceedings of the National Academy of Sciences*, 1998.

[43] M. Belkin and P. Niyogi, "Laplacian eigenmaps for dimensionality reduction and data representation," *Neural computation*, vol. 15, no. 6, pp. 1373–1396, 2003.

[44] X. He and P. Niyogi, "Locality preserving projections," in *Advances in neural information processing systems*, 2004, pp. 153–160.

[45] M. M. Crawford, L. Ma, and W. Kim, "Exploring nonlinear manifold learning for classification of hyperspectral data," in *Optical Remote Sensing*. Springer, 2011, pp. 207–234.

[46] M. Sugiyama, "Dimensionality reduction of multimodal labeled data by local fisher discriminant analysis," *Journal of machine learning research*, vol. 8, no. May, pp. 1027–1061, 2007.

[47] D. Lunga, S. Prasad, M. M. Crawford, and O. Ersoy, "Manifold-learning-based feature extraction for classification of hyperspectral data: A review of advances in manifold learning," *IEEE Signal Processing Magazine*, vol. 31, no. 1, pp. 55–66, January 2014.

[48] P. Goyal and E. Ferrara, "Graph embedding techniques, applications, and performance: A survey," *CoRR*, 2017.

[49] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *ICLR 2017*, 2017.

[50] J. L. W. L. Hamilton, R. Ying, "Inductive representation learning on large graphs," in *Advances in Neural Information Processing Systems*, 2017.

[51] Y. Li, D. Tarlow, M. Brockschmidt, and R. Zemel, "Gated graph sequence neural networks," *arXiv preprint arXiv:1511.05493*.

[52] Y. L. M. Henaff, J. Bruna, "Deep convolutional networks on graph-structured data," *arXiv preprint arXiv:1506.05163*.

[53] D. K. Duvenaud, D. Maclaurin, J. Iparraguirre, R. Bombarell, T. Hirzel, A. Aspuru-Guzik, and R. P. Adams, "Convolutional networks on graphs for learning molecular fingerprints," in *Advances in neural information processing systems*, 2015, pp. 2224–2232.

[54] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," *arXiv preprint arXiv:1312.6203*, 2014.

[55] P. V. M. Defferrard, X. Bresson, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Advances in Neural Information Processing Systems*, 2016.

[56] F. R. Chung and F. C. Graham, *Spectral graph theory*. American Mathematical Soc., 1997, no. 92.

[57] D. K. Hammond, P. Vandergheynst, and R. Gribonval, "Wavelets on graphs via spectral graph theory," *Applied and Computational Harmonic Analysis*, vol. 30, no. 2, pp. 129–150, 2011.

[58] M. Henaff, J. Bruna, and Y. LeCun, "Deep convolutional networks on graph-structured data," *arXiv preprint arXiv:1506.05163*.

[59] J. R. Stevens, R. G. Resmini, and D. W. Messinger, "Spectral-density-based graph construction techniques for hyperspectral image analysis," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 10, pp. 5966–5983, 2017.

[60] Y. Zhang, H. L. Yang, D. Lunga, S. Prasad, and M. Crawford, "Spatial context driven manifold learning for hyperspectral image classification," in *2014 6th Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS)*, June 2014, pp. 1–4.

[61] H. Wu and S. Prasad, "Semi-supervised dimensionality reduction of hyperspectral imagery using pseudo-labels," *Pattern Recognition*, vol. 74, pp. 212–224, 2018.

[62] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[63] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[64] S. Berisha, M. Lotfollahi, J. Jahanipour, I. Gurcan, M. Walsh, R. Bhargava, H. Van Nguyen, and D. Mayerich, "Deep learning for ftir histology: leveraging spatial and spectral features with convolutional neural networks," *Analyst*, vol. 144, no. 5, pp. 1642–1653, 2019.

[65] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[66] H. Ni, Z. Feng, Y. Guan, X. Jia, W. Chen, T. Jiang, Q. Zhong, J. Yuan, M. Ren, and X. Li, "Deepmapi: a fully automatic registration method for mesoscopic optical brain images using convolutional neural networks," *Neuroinformatics*, vol. 19, no. 2, pp. 267–284, 2021.

[67] X. Wang, W. Zeng, X. Yang, Y. Zhang, C. Fang, S. Zeng, Y. Han, and P. Fei, "Bi-channel image registration and deep-learning segmentation (birds) for efficient, versatile 3d mapping of mouse brain," *Elife*, vol. 10, p. e63455, 2021.

[68] H. Ni, C. Tan, Z. Feng, S. Chen, Z. Zhang, W. Li, Y. Guan, H. Gong, Q. Luo, and A. Li, "A robust image registration interface for large volume brain atlas," *Scientific reports*, vol. 10, no. 1, pp. 1–16, 2020.

[69] C. Tan, Y. Guan, Z. Feng, H. Ni, Z. Zhang, Z. Wang, X. Li, J. Yuan, H. Gong, Q. Luo, and A. Li, "Deepbrainseg: Automated brain region segmentation for micro-optical images with a convolutional neural network," *Frontiers in Neuroscience*, vol. 14, p. 179, 2020. [Online]. Available: https://www.frontiersin.org/article/10.3389/fnins.2020.00179

[70] F. L. Bookstein, "Thin-plate splines and the atlas problem for biomedical images," in *Information Processing in Medical Imaging*, A. C. F. Colchester and D. J. Hawkes, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 1991, pp. 326–342.

[71] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," *CoRR*, vol. abs/1505.04597, 2015. [Online]. Available: http://arxiv.org/abs/1505.04597

[72] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[73] A. A. Bastidas and H. Tang, "Channel attention networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2019, pp. 0–0.

[74] A. Van Etten, D. Lindenbaum, and T. M. Bacastow, "Spacenet: A remote sensing dataset and challenge series," *arXiv preprint arXiv:1807.01232*, 2018.

[75] L.-C. Chen, Y. Yang, J. Wang, W. Xu, and A. L. Yuille, "Attention to scale: Scale-aware semantic image segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 3640–3649.

[76] J. S. Lim, "Two-dimensional signal and image processing," *Englewood Cliffs*, 1990.

[77] J. Canny, "A computational approach to edge detection," *IEEE Transactions on pattern analysis and machine intelligence*, no. 6, pp. 679–698, 1986.

[78] G. Csurka, D. Larlus, F. Perronnin, and F. Meylan, "What is a good evaluation measure for semantic segmentation?." in *Bmvc*, vol. 27, no. 2013, 2013, pp. 10–5244.

[79] D. R. Martin, C. C. Fowlkes, and J. Malik, "Learning to detect natural image boundaries using local brightness, color, and texture cues," *IEEE transactions on pattern analysis and machine intelligence*, vol. 26, no. 5, pp. 530–549, 2004.

[80] K. E. Iverson, "A programming language," in *Proceedings of the May 1-3, 1962, spring joint computer conference*, 1962, pp. 345–351.

[81] J. Schindelin, I. Arganda-Carreras, E. Frise, V. Kaynig, M. Longair, T. Pietzsch, S. Preibisch, C. Rueden, S. Saalfeld, and B. Schmid, "Fiji: an open-source platform for biological-image analysis," *Nature methods*, vol. 9, no. 7, pp. 676–682, 2012.