Silesian University of Technology

Doctoral Dissertation

Unmixing of Hyperspectral Images: Where Artificial Intelligence Meets Earth Observation

Author:
Lukasz Tulczyjew, MSc

Supervisor:
Jakub Nalepa, PhD, DSc

A thesis submitted in fulfillment of the requirements
for the degree of Doctor of Philosophy
in the

Department of Algorithmics and Software,
Silesian University of Technology

July 9, 2025

Declaration of Authorship

I, Lukasz Tulczyjew, MSc, declare that this thesis titled, "Unmixing of Hyperspectral Images: Where Artificial Intelligence Meets Earth Observation" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

SILESIAN UNIVERSITY OF TECHNOLOGY

Abstract

Faculty of Automatic Control, Electronics and Computer Science
Department of Algorithmics and Software

Doctor of Philosophy

Unmixing of Hyperspectral Images: Where Artificial Intelligence Meets Earth Observation

by Lukasz Tulczyjew, MSc

This doctoral thesis focuses on improving the analysis of hyperspectral images by addressing key challenges across the entire machine learning workflow, from data preparation to model deployment. The research begins by tackling the problem of biased model evaluation, often caused by spatial correlations in hyperspectral imagery. A rigorous data splitting and validation strategy is introduced to prevent information leakage between training and test sets, ensuring fair and trustworthy assessment of model performance.

The next part of the work investigates how the size of the training dataset influences the accuracy and reliability of material abundance estimations. Through systematic experiments, this study highlights the sensitivity of deep learning models to the availability of labeled data, offering valuable insights for designing robust models in data-scarce environments.

Building on these foundations, the thesis benchmarks state-of-the-art deep learning methods for hyperspectral image analysis and proposes new model architectures designed to improve unmixing accuracy. These include deep ensemble techniques, graph convolutional neural networks combining spectral and spatial features as well as attention modules, and a flexible band selection method that reduces data complexity without compromising performance.

Finally, the research addresses the practical challenge of deploying these advanced models in real-world settings, particularly on resource-limited platforms such as satellites. To support this, the thesis explores model quantization techniques that significantly reduce computational demands, making it possible to run complex deep learning models efficiently on edge devices.

Together, these contributions form a complete, end-to-end framework for hyperspectral image analysis — from data preparation and feature selection to model development, evaluation, optimization, and deployment. This work not only advances the scientific understanding of hyperspectral unmixing but also delivers practical tools and methods for operational Earth observation applications.

Acknowledgements

I would like to express my heartfelt gratitude to my supervisor, Jakub Nalepa, PhD, DSc, for his invaluable guidance, support, and encouragement throughout the course of this research. His expertise has been instrumental from the very beginning of my academic journey, helping me develop as a researcher over eight years of collaboration during which we have worked together on multiple publications. His insightful advice, his constant availability for discussion, and the trust he placed in me have shaped both this thesis and my development as a researcher. I am truly grateful for the opportunity to work under his exceptional supervision.

Contents

List of Figures

List of Tables

List of Abbreviations

AE	Autoencoder
AI	Artificial Intelligence
CB	Cube Based
ChebConv	Chebyshev Graph Convolutions
CNN	Convolutional Neural Network
DCAE	Deep Convolutional Autoencoder
DL	Deep Learning
DT	Decision Tree
EV	Explained Variance
FPGA	Field-Programmable Gate Arrays
GAwLL	Genetic Algorithm with Linkage Learning
GCNN	Graph Convolutional Neural Network
GCNN-Cheb	Graph Convolutional Neural Network with Chebyshev Polynomial
GRU	Gated Recurrent Unit
GSD	Ground Sampling Distance
GSV	Geospatially Stratified Validation
HSI	Hyperspectral Image
HSS	Hyperspectral Segmentation
HU	Hyperspectral Unmixing
IQR	Inter-Quartile Range
JR	Jasper Ridge
KDE	Kernel Density Estimation
kNN	K-Nearest Neighbor
LMM	Linear Mixing Model
LReLU	Leaky Rectified Linear Unit
LSTM	Long Short-Term Memory
M	Mean
MAE	Mean Absolute Error
MB	Multibranch
MinMax	Minimum and Maximum
ML	Machine Learning
MLP	Multilayer Perceptron
MSE	Mean Squared Error
NMF	Nonnegative Matrix Factorization
OA	Overall Accuracy

PB	Pixel Based
PPI	Pixel Purity Index
PReLU	Parametric Rectified Linear Unit
PT	Pre-Trained
PTQ	Post-Training Quantization
QAT	Quantization-Aware Training
ReLU	Rectified Linear Unit
RF	Random Forest
RMSE	Root Mean Squared Error
RMSAAD	Root Mean Squared Abundance Angle Distance
rmsAAD	Root Mean Squared Abundance Angle Distance
RNN	Recurrent Neural Network
RS	Random Sampling
RS	Remote Sensing
Sa	Samson
SAD	Spectral Angle Distance
SAM	Spectral Angle Mapper
SID	Spectral Information Divergence
SOTA	State of the Art
SVR	Support Vector Regression
tanh	Hyperbolic Tangent
TL	Transfer Learning
Ur	Urban
VAE	Variational Autoencoder
VIG	Variable Integration Graph
wVIG	Weighted Variable Integration Graph
XAI	Explainable AI
κ	Kappa Coefficient

Dedicated to my parents, Adam and Elżbieta, my grandmother, Janina, and my wife, Natalia.

Chapter 1

Introduction

1.1 Introduction to Hyperspectral Image Analysis

Hyperspectral image (HSI) analysis is a pivotal area in the field of remote sensing, focusing on extracting meaningful information from captured HSIs. An HSI consists of a continuous spectrum of electromagnetic data represented as a series of images, referred to as *bands*. Each band captures a narrow range of the spectrum, providing detailed reflectance characteristics of the scene's objects, enabling analysis based on spectral signatures [5], [125], [136].

Despite the richness of hyperspectral information, HSI analysis poses several inherent challenges that hinder its direct and efficient application. Firstly, HSIs are characterized by extremely high dimensionality—often encompassing hundreds of contiguous spectral bands—which leads to massive data volumes that are computationally expensive to store, transmit, and process [46], [52], [227], [241]. This is especially critical in satellite-based scenarios, where data downlink capacity is limited. As a result, efficient techniques such as smart compression, including band selection and feature extraction, become essential to reduce data size while preserving critical information [150], [221], [241]. Band selection, in particular, not only enhances computational efficiency but also promotes interpretability of the analysis results by retaining only the most informative and non-redundant bands. Another major challenge is the lack of labeled data: while raw hyperspectral images are abundant, annotated datasets are scarce due to the high cost and complexity of ground-truth acquisition. This scarcity significantly limits the development and scalability of supervised learning models. Additionally, factors such as high spectral similarity between different materials, spectral mixing, and external conditions like cloud coverage further complicate the analysis [1], [251]. Preprocessing steps such as cloud detection and masking can help reduce the volume of irrelevant or corrupted data and improve the reliability of downstream tasks [272]. Collectively, these challenges necessitate the development of robust, data-efficient, and interpretable machine learning (ML) methods for HSI processing, a topic that has gained increasing attention in recent literature [194], [215].

Processing of HSIs often involves *segmentation* (HS), i.e., a pixel-level classification. It is a critical step utilized across numerous scientific and industrial applications, such as mineralogy [121], agriculture [79], medicine [153], forensics [57], and others [39], [154], [205]. The primary aim of segmentation is to partition the input image's pixels into non-overlapping groups, each corresponding to a distinct class of objects. However, the high dimensionality of HSIs, often comprising hundreds of bands, poses significant challenges to the segmentation process. Prior research [151] has demonstrated that dimensionality reduction through selective band selection can maintain segmentation performance. Additionally, the inherent similarity of intensity values across materials or their intermixing complicates pixel grouping into correct groups. To address these challenges, various machine learning approaches for HSI segmentation have been proposed, which will be explored in detail in Chapter 4.

In addition to spectral similarity issues, HSIs often suffer from low spatial resolution due to large ground sampling distances (GSDs). This limitation causes individual pixels to represent mixtures of multiple materials, further complicating the segmentation process. When such mixed pixels occur, the model must infer the predominant and secondary materials contributing to the observed spectra. This introduces the challenge of *hyperspectral unmixing* (HU), which aims to estimate the abundance of individual materials, known as endmembers, within each pixel [89]. HU is crucial in various scientific and operational domains, including precision agriculture (e.g., crop type monitoring and stress detection) [152], mineral exploration (e.g., mapping lithological units) [20], [209], urban planning (e.g., land cover classification) [30], and environmental monitoring (e.g., pollutant dispersion or vegetation composition analysis) [100], [191]. Moreover, the domains previously referenced in the context of HS also benefit from HU techniques. Mathematically, HU can be conceptualized as a multi-target regression problem, where each target corresponds to the abundance of a specific endmember in the image.

In this dissertation, we focus on the HU task to address several key challenges in the field. Our primary objectives are to overcome the issue of limited availability of labeled data and to develop artificial intelligence-based models that can be effectively optimized even with small training sets. We aim to tackle the problem of mixed spectral signatures by proposing multiple deep learning (DL) approaches, including convolutional neural networks and graph-based methods. Furthermore, we investigate the influence of training dataset size on model performance, providing a comprehensive evaluation. Moreover, we focus on establishing non-overlapping training sets, without any information leak between training and testing samples, which is a common problem in HSI analysis as the task is bounded by a single scene. An additional aspect of our work involves exploring deployment on edge devices such as satellite sensors. In this context, we examine the impact of model quantization and its implications for real-world applications. Overall, this research contributes to advancing the state of the art and pushing the boundaries of scientific progress in this domain.

1.2 Problem Analysis

With the rapid growth of remote sensing data acquisition in recent years, there is an increasing demand for high-quality HU models. However, constructing such models requires access to accurate ground-truth (GT) information, which is costly, time-intensive, and prone to human errors. Labeling hyperspectral images is both costly and time-intensive, as it involves assigning each pixel a real-valued vector indicating the abundance fractions of endmembers [17], [20], [89], [191]. Needless to say such vectors have to be precise to enable real use case possibility.

This task becomes even more demanding due to the large number of pixels in hyperspectral images, which directly increases the volume of data to be processed and raises the computational cost of the models. The spectral mixing phenomenon further complicates the labeling process, as pixels often represent mixtures of multiple materials [20]. Additional challenges arise during the data acquisition phase, such as variations in illumination conditions and the presence of atmospheric noise, which further degrade the quality of the data [19], [100], [250].

These issues not only affect the labeling process but also impact both segmentation and unmixing tasks in hyperspectral image analysis. To address these challenges, various methods have been proposed in the literature, for the complete description of the literature, please refer to Chapter 4—we discuss there different approaches that focus on HU as well as the segmentation task.

The task of HU can be split into two main approaches: (*i*) Linear Mixing Model (LMM), and (*ii*) Non-linear Mixing Model (NMM). The former method assumes that the observed spectral signal of a pixel is created by a linear combination of the endmembers spectra and the abundances of each respective target in the HSI [93]. The formula can be given as:

$$\mathbf{p} = \Sigma_{i=1}^{k} a_i \mathbf{e_i} + \epsilon, \tag{1.1}$$

where $\mathbf{p}$ is the observed signal of a pixel as a spectral vector $p := [p_1, p_2, ..., p_b]^T$, a_i denotes the abundance of endmember indexed by i, $\mathbf{e_i}$ is the endmember spectrum, and ϵ represents the noise attributed to the image, for instance the additive noise.

The components denoted as a_i for each pixel compose a vector with k elements. Such vector has two constraints in the problem of HU. The first one is the *abundance nonnegativity constraint*, which dictates that all percentages of abundances cannot be lower than zero. Furthermore, the *abundance sum constraint* requires the sum of the entire vector for all pixels to be equal to 1.

Though the model itself is relatively simple and offers acceptable approximation for scattered spectra in many real scenarios, such model has its limitations in situations when the collected light is refracted or disturbed by various materials, difficult acquisition conditions or any medium such as water or atmospheric compositions. The assumption that all interactions among endmembers are linear usually does not allow to perform satisfactory unmixing.

To overcome such problems the NMM model accounts for interaction between endmembers that are outside the span of linear unmixing methods. It incorporates but is not limited to effects such as material contamination, noise atmospheric conditions, shading, overlapping spectra,

The NMM incorporates *bilinear models* which are designed to address the nonlinearity inherently present in many real-world applications, where complex varying illumination conditions and material interactions on microscopic level hinder accurate HU. The methods encapsulate constrained nonnegative matrix factorization techniques and its modifications and generalized bilinear models [89], for complete overview of different HU techniques, please see Chapter 4.

Chapter 2

Contribution

This doctoral thesis focuses on improving the analysis of hyperspectral images by addressing key challenges across the entire machine learning workflow, from data preparation to model deployment. The research tackles critical issues in HU and broader HSI analysis, including unbiased validation, the scarcity of labeled training data, and the development of effective algorithms for feature extraction and dimensionality reduction. The included works collectively address these challenges through innovative approaches, contributing significantly to the field and advancing its practical applications.

The research begins by tackling the problem of biased model evaluation, often caused by spatial correlations in hyperspectral imagery. The first contribution introduces a method to combat information leakage by ensuring clear separation between training and test data. This work proposes a rigorous data splitting and validation strategy that prevents information leakage between training and test sets, avoiding overly optimistic performance assessments and setting a new standard for evaluating unmixing techniques. This approach ensures fair and trustworthy assessment of model performance.

Another major focus addresses the challenge of limited labeled data for supervised learning in hyperspectral unmixing. The impact of training set size on the quality of abundance estimation is carefully analyzed through systematic experiments. This study highlights the sensitivity of deep learning models to the availability of labeled data, providing insights into the robustness of these models and their dependency on data availability. The findings offer valuable guidance for designing robust models in data-scarce environments.

Building upon advances in deep learning, this research explores the potential of deep ensembles, which combine the strengths of multiple convolutional models. This approach enhances hyperspectral data unmixing performance by integrating supervised fusing methods and proposing a novel augmentation technique. The result is a robust framework that outperforms traditional methods and provides a scalable solution for handling hyperspectral imagery.

Further advancing the field, this thesis benchmarks state-of-the-art deep learning methods for hyperspectral image analysis and proposes new model architectures

designed to improve unmixing accuracy. A recap of multibranch convolutional neural networks from previous PhD Candidate research is presented, which integrates spectral, spatial, and spectral-spatial features for unmixing tasks. These multibranch architectures are treated as state-of-the-art approaches in the field. **Note:** The work on multi-branch CNNs has started while working on the PhD Candidate's Master's thesis, and has become a point of departure to the works presented in this dissertation.

Furthermore, this work utilizes graph convolutional neural networks (GCNNs) to tackle unmixing challenges, comparing their performance against established multibranch models and other literature-based state-of-the-art approaches. The multibranch architecture demonstrates superior performance in abundance estimation via the fusion of spectral and spectral-spatial features extracted via 1D, 2D, and 3D convolutional branches, yielding statistically significant improvements in root mean square error (RMSE) and root mean square abundance angle distance (rmsAAD) metrics across multiple datasets. However, ablation studies revealed that the combination of 1D and 3D branches provides the most significant improvements, while the 2D branch addition offers no statistically distinguishable benefits due to redundancy and insufficient spatial discrimination. Notably, GCNNs achieve comparable HU performance to state-of-the-art architectures despite having significantly fewer trainable parameters and simpler architectural design. Beyond their effectiveness in unmixing, GCNNs demonstrate marked advantages in training efficiency, converging more rapidly and requiring substantially less training time compared to deep, multi-branch counterparts. This computational efficiency, combined with their lower architectural complexity, positions GCNNs as particularly attractive for deployment on resource-constrained platforms such as satellites. Additionally, the interpretability of GCNN architectures offers unique advantages for explainable AI approaches in remote sensing, while their robustness to noise and deployment versatility make them highly suitable for practical applications in operational and edge environments.

Additionally, this work introduces CANNIBAL, a novel non-parametric feature selection method that addresses the critical challenge of hyperspectral data dimensionality. This method leverages unsupervised clustering of inter-band dependencies to identify most informative bands, significantly reducing data complexity without sacrificing model quality. Statistical analysis through pairwise Wilcoxon tests over benchmark datasets confirmed that configurations with more than 25 CANNIBAL-selected bands perform statistically equivalent to full-spectrum analysis, enabling up to 85% data reduction (from 162 to 25 bands in the utilized Urban benchmark dataset) while eliminating redundant spectral information. The method consistently outperforms other feature selection approaches with statistical significance while requiring significantly fewer features, parameters, and demonstrating substantial practical value for computational efficiency and storage optimization in operational hyperspectral systems.

Finally, this work addresses the practical deployment challenges of deep neural

networks through quantization analysis. The research evaluates quantization capabilities with real-life data, examining how quantization affects model accuracy, latency, and memory footprint under practical deployment scenarios. The quantization approach achieves substantial storage savings, reducing the full-precision model from 25 MB to 2.5 MB with employed optimization (90% overall reduction). Such significant compression enables efficient deployment on satellite systems and embedded platforms with limited memory and bandwidth.

Overall, these contributions form a complete, end-to-end framework for hyperspectral image analysis—from data preparation and feature selection to model development, evaluation, optimization, and deployment. This dissertation emphasizes the importance of developing techniques that are not only accurate and robust but also practical for real-world Earth observation scenarios. This body of work demonstrates the field's evolution, bridging theoretical advancements with applied solutions while addressing critical deployment considerations through quantization and model optimization techniques. The research not only advances the scientific understanding of hyperspectral unmixing but also delivers practical tools and methods for operational Earth observation applications.

2.1 Published Works Included in This Thesis

Below, all of the published articles that are related and included in this thesis are listed coupled with short description of each publication:

- **Lukasz Tulczyjew**, Michal Kawulok, Nicolas Longépé, Bertrand Le Saux, and Jakub Nalepa. "Unbiased Validation of Hyperspectral Unmixing Algorithms." In IGARSS 2023-2023 IEEE International Geoscience and Remote Sensing Symposium, pp. 7344-7347. IEEE, 2023 [235].
  CORE ranking: C.

  This article tackles the information leakage due to spatial correlations in an input HSI, compromising the validation of techniques using spatial information. To address this, a new algorithm is proposed for unbiased validation, ensuring separate training and test samples. Experiments reveal that random sampling overestimates algorithm performance, highlighting the importance of this rigorous validation method.

  In this work, we focus on establishing non-overlapping training and testing data splits over a single hyperspectral image. This is of critical importance, as incorrectly designed validation procedures may easily lead to overly optimistic estimations of the model's generalization capabilities, as shown in [176]. The proposed algorithm has three additional advantages: *i*) it is easily parameterizable, i.e., allows us to specify a variable size for training and test splits, *ii*) it incorporates the possibility of generating multiple folds to simulate cross-validation scenario, and *iii*) it allows us to sample training and test splits encapsulating spectral as well as spatial features by pixel-wise and patch-wise sampling. In Figure 2.1, we present the visualization of such sampling strategy with pixel-wise approach.

FIGURE 2.1: Visualization of training (green shade) and test subsets sampled from the Urban HU scene. The region where the borders between training and testing sets meet are magnified. Those edge pixels (in violet and red) are discarded and not included in neither set, as they would induce information leak. This figure was included in our publication in [235].

In our experiments, we utilized three widely-researched HU benchmark datasets, i.e., Samson (Sa, 95×95, containing 156 bands), Urban (Ur, 207×307, 162 bands) and Jasper Ridge (JR, 100×100, 198 bands) [287]. The Sa dataset encapsulates three endmembers: #1 Soil, #2 Tree and #3 Water, for Ur six endmembers are incorporated: #1 Asphalt, #2 Grass, #3 Tree, #4 Roof, #5 Metal, and #6 Dirt, finally, for JR, four targets are utilized: #1 Road, #2 Water, #3 Soil, and #4 Tree. In the experiments, we incorporate 30-fold Monte Carlo cross-validation method, and sample 30 test sets that *do not change* with the variation of the training set size.

From the experiments, we conducted that random sampling results are overly optimistic for all benchmark datasets. Furthermore, the errors obtained for our proposed non-overlapping sampling went up by around 550% larger than for the formerly mentioned method showing that there indeed exist an information leak between the training and testing sets. Such leakage compromises the validity of the results and undermines the effectiveness of the method as a reliable benchmarking tool.

Contribution of the PhD candidate:

The PhD Candidate was responsible for executing the literature review, conducting the experiments required in this work, and (together with other co-authors) worked on preparation of the manuscript including visualizations, tables and descriptions of the results.

- **Lukasz Tulczyjew** and Jakub Nalepa. "Investigating the impact of the training set size on deep learning-powered hyperspectral unmixing." In 2021 IEEE International Geoscience and Remote Sensing Symposium IGARSS, pp. 2024-2027. IEEE, 2021 [238].
  CORE ranking: C.

  This work addresses the problem of limited labeled training data for supervised HU. In particular, we investigate how the size of the training dataset affects the performance of DL models for HU. Our experiments focus on understanding and quantifying the impact of training set size on the abundance maps extracted by DL models. This aspect is pivotal in practical supervised learning for hyperspectral data analysis, as capturing high-quality GT labels is extremely challenging. Acquiring GT data typically requires labor-intensive field campaigns involving precise geo-referencing, expert annotations, and often the use of sophisticated and costly instruments. Moreover, the annotation process is prone to human error and subjectivity, particularly when it involves manual interpretation of complex spectral signatures. In many real-world scenarios, environmental variability and limited accessibility of regions of interest further complicate reliable data collection. As a result, ground truth datasets for hyperspectral unmixing are often limited in size and may not fully capture the variability present in the scene, which can significantly hinder the training and generalization of deep learning models. Therefore, understanding how the size and quality of the training dataset influence model performance is crucial for designing practical and effective HU solutions. Using two conceptually different architectures—a Convolutional Neural Network (CNN) and a Deep Convolutional Autoencoder (DCAE)—we investigate how varying the size of training datasets impacts unmixing accuracy.

  From our experimental study we conducted the key findings as following:

  - Very small training datasets (amounting to $\leq$6% training examples of total available data) result in poor unmixing performance.
  - Increasing the dataset size significantly improves performance up to a magnitude of 33% available data, beyond which further improvements plateau in terms of quality of the abundance estimation.
  - Cube-based models (which incorporate spatial information) generally outperform pixel-based models while performing HU.

The publication employs established hyperspectral datasets (JR and Ur) and quantitative metrics like RMSE defined in Equation 2.1 and rmsAAD formulated in Equation 2.2, to evaluate model performance. The findings emphasize the need for cautious dataset preparation and suggest that optimal training data sizes can mitigate the costly manual annotation process without sacrificing model quality.

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{N}(\boldsymbol{a}_i - \hat{\boldsymbol{a}}_i)^2}{N}}, \tag{2.1}$$

where N is the number of pixels in the test set, $\boldsymbol{a}$ and $\hat{\boldsymbol{a}}$ represent the true and predicted unmixing vectors. The rmsAAD measure is:

$$\text{rmsAAD} = \sqrt{\frac{\sum_{i=1}^{N} \arccos\left(\frac{\boldsymbol{a}_i^\top \hat{\boldsymbol{a}}_i}{\|\boldsymbol{a}_i\|\|\hat{\boldsymbol{a}}_i\|}\right)^2}{N}}. \tag{2.2}$$

TABLE 2.1: The results of the Friedman statistical tests followed by the Dunn's multiple comparisons tests for (a) Jasper Ridge and (b) Urban, obtained for rmsAAD. We indicate the pairs (a–b, where a and b are the percentage of all training pixels, in %) for which the means are significantly different (at $p < 0.05$). This table was also utilized in the PhD Candidate's work in [238].

Variant	Set	Pairs
Pixel-based CNN	(a)	1–13, 1–33, 1–66, 1-100, 6–33, 6–66, 6–100, 13–66, 13–100
	(b)	1–33, 1–66, 1–100, 6–66, 6–100, 13–100
Cube-based CNN	(a)	1–13, 1–33, 1–66, 1–100, 6–33, 6–66, 6–100, 13–66, 13–100
	(b)	1–33, 1–66, 1–100, 6–66, 6–100, 13–66, 13–100
Pixel-based DCAE	(a)	—
	(b)	1–33, 1–66, 1–100
Cube-based DCAE	(a)	1–13, 1–33, 1–66, 1–100, 6–66, 6–100
	(b)	1–100

To statistically assess the influence of training set size on model performance, we applied the Friedman test followed by Dunn's post-hoc multiple comparisons test on the rmsAAD metric. These non-parametric tests allow us to evaluate whether the differences in performance across different training set sizes are statistically significant. The results, summarized in Table 2.1, highlight which pairs of training set sizes (expressed as percentages of all available pixels) yield significantly different results at a confidence level of $p < 0.05$. Notably, for both Jasper Ridge and Urban scenes, the pixel-based and cube-based CNN models exhibit significant differences across a wide range of training percentages, particularly when comparing the smallest training sizes (e.g., 1%) to larger ones (e.g., 66% or 100%). This indicates a strong dependency of CNN-based models on the volume of training data. In contrast, the DCAE models, especially the pixel-based variant, show fewer significant pairwise differences, suggesting a higher level of robustness to limited training data. These findings reinforce the importance of dataset size in hyperspectral unmixing and underline how different deep learning architectures respond to training data scarcity.

This research provides valuable insights, e.g., minimal magnitudes of training sets for the most well established HU benchmark datasets, for training DL architectures in scenarios with limited labeled data, especially in real-world hyperspectral imaging applications.

Contribution of the PhD Candidate:

The PhD Candidate performed the literature review, preparation of datasets and experimental setup, performed the experiments and analyzed the results. Together with the co-authors, the PhD Candidate prepared the manuscript including the visualizations and tables.

- Jakub Nalepa, Michal Myller, **Lukasz Tulczyjew**, and Michal Kawulok. "Deep ensembles for hyperspectral image data classification and unmixing." Remote Sensing 13, no. 20 (2021): 4133 [178].
  Impact factor: 5.349. Ministerial score (points): 100.

  This study introduces deep ensembles combining advancements in convolutional models with a supervised fuser for output aggregation. A novel model augmentation technique is also proposed, injecting Gaussian noise into model weights to create new networks. Experiments demonstrate that these deep ensembles outperform traditional models and fusion methods in hyperspectral classification and unmixing.

  In this work, we introduced a new deep ensemble methods that draw upon the strengths of multiple convolutional architectures for hyperspectral segmentation and unmixing. Furthermore, we proposed a supervised fuser, which accurately merges the outputs from these individual base learners. Also, a new model augmentation procedure in which Gaussian noise is injected into the weights of the models was introduced. This approach enables the synthesis of additional network instances derived from the original model, without compromising its essential characteristics—such as its learned feature representations, spectral-spatial sensitivity, and overall predictive behavior. These core properties ensure that the augmented models remain functionally similar to the original, allowing for the generation of meaningful ensemble predictions or robustness analysis while preserving the integrity of the original model's knowledge.

  We carried out extensive experiments concerning both classification and unmixing tasks and observed that our deep ensembles consistently outperform standard spectral or spectral-spatial deep networks. We also compared our approach to traditional ensembles that employ voting or averaging and found that our supervised fusion strategy offers marked significant improvements in hyperspectral image analysis.

  In this research our contribution incorporates heterogeneous ensemble methods, which encapsulate Convolutional Neural Networks (CNNs) with both, spectral and spatial dimensions to tackle the problems of hyperspectral segmentation

and unmixing. In the task of hyperspectral classification our base models constitute spectral 1D CNN network, a 2.5D-CNN architecture [69], and 3D-CNN model [180]. The former method performs pixel-wise classification, whereas the last two models incorporate the spatial neighborhood context into the input features, hence allowing us to obtain patch-wise prediction. The distinction between the last two models is that the 2.5D architecture extends its kernel throughout the entire electromagnetic spectrum and the 3D model has a fixed filter length in all dimensions equal to 3, thus allowing us to extract fine-grained spectral relations, based on the receptive field of its kernels. In the problem of HU, we operate on pixel- and patch-wise paradigms and utilize models introduced in [112], [278], both the pixel- and cube-based variants of DCAEs. It is worth emphasizing that all models could be decomposed into two underlying parts, i.e., the feature extractor which incorporates convolutional layers, and the predictor module, which is responsible of transforming the activation maps into interpretable output that is problem specific. For classification the number of output neurons is equal to the number of classes, whereas for unmixing, the number of endmembers dictate the size of output activations.

Our thorough experimental process and findings prove that employed models allow us to improve existing State-Of-The-Art (SOTA) models and other classical ensemble methods. Finally, we provide a clear and open-source format of our development and our algorithms are publicly available at `https://github.com/ESA-PhiLab/hypernet/tree/master/beetles`.

In our experimental findings, we observed that the spectral 1D-CNN consistently achieved higher classification accuracy and outperformed both the 2.5D-CNN and 3D-CNN models. We attribute this finding to the scarcity and limited diversity of training samples, which can lead large-capacity networks to overfit and thus not generalize effectively, especially considering noise injection to the trained models. We further noted that if the training set size becomes larger, a spectral-spatial 3D-CNN is capable of markedly surpassing the 1D-CNN approach by leveraging both local spatial neighborhoods and rich spectral information for accurate class identification. We also recognized that the obstacle of insufficient labeled data can be addressed with a variety of established strategies, including transfer learning and data augmentation, as these methods have proven particularly effective in earlier research [123].

In the HU experiments, we observed that expanding the training sets generally strengthens the performance of DCAEs, yet only marginal improvements occur once we move beyond a certain threshold (i.e., when using the entire sets or 66% randomly selected samples). We also found that constructing heterogeneous ensembles enhances unmixing for all considered training sizes, with the Random Forest (RF) fuser consistently achieving the best results. We also noted that increasing the size of the training data mitigates the limitations of weaker models

trained on small ground-truth sets. Moreover, our experimental findings confirm that the proposed ensembles outperform standard hyperspectral unmixing techniques.

Because DCAE is an unsupervised approach, we concluded that creating augmented versions of DCAEs would be less beneficial, as there is no straightforward way to utilize supervised fusers for integrating the predictions of both unaltered and perturbed unsupervised models. Our initial tests also implied that averaging the outputs of the original and noise-injected 1D-DCAEs did not yield observable enhancements.

Nonetheless, injecting Gaussian noise proved to substantially boost the generalization abilities of 1D-CNNs for both analyzed datasets, allowing us to construct high-performing ensembles from relatively weak individual models. We believe that this type of ensemble learning—coupled with our model augmentation technique—could accelerate the application of CNN-based unmixing methodologies in scenarios where the availability of labeled training data is severely limited or collecting new examples is impractical.

Contribution of the PhD Candidate:

The PhD Candidate was responsible for all aspects of the hyperspectral unmixing research, including implementing SOTA models from literature, developing and applying noise augmentation techniques, training all unmixing models, conducting all experiments, collecting and analyzing results, and preparing the manuscript sections related to hyperspectral unmixing. The PhD Candidate also contributed to the literature review, dataset preparation, experimental setup design, and collaborated with co-authors on manuscript preparation including visualizations and tables for the complete study.

- **Lukasz Tulczyjew**, Michal Kawulok, Nicolas Longépé, Bertrand Le Saux, and Jakub Nalepa. "A multibranch convolutional neural network for hyperspectral unmixing." IEEE Geoscience and Remote Sensing Letters 19 (2022): 1-5 [233]. Impact factor: 4.0. Ministerial score (points): 140.

  Deep Machine Learning Models for HU:

  This work addresses the challenge of HU, a critical task in analyzing hyperspectral data. Building on the success of deep learning, which has outperformed traditional methods and is suitable for deployment on Earth observation satellites, we propose a multibranch convolutional neural network that integrates spectral, spatial, and spectral-spatial features. Experimental results, supported by an ablation study, show that our approach surpasses existing techniques, achieving superior fractional abundance estimation. Additionally, we examined the impact of reduced training sets on algorithm performance and robustness to noise, recognizing the practical difficulties of obtaining extensive ground-truth datasets in emerging Earth observation scenarios. To tackle the task of HU

in our approach we utilize the concept of multibranch architectures. We utilize different combinations of fusing features based on three possible domains: spectral, spatial and spectral-spatial.

In this work, we investigate the possibility of utilizing convolutional layers and fuse spectral, spatial and spectral-spatial features to tackle the task of HU. In our experiments, we proved that incorporating such methodology allows us to outperform other SOTA models and provide high quality fractional abundances. Furthermore, we analyzed the influence of varying training set sizes on the generalization capabilities of DL models and the impact of noise injection on the robustness of such methods.

The first variation incorporates three 1D convolutional layers and max pooling operations with only spectral features and performs pixel-wise prediction. Subsequently, the 2D model incorporates patch-wise input and encapsulates five 3-dimensional kernels, spanning across the entire spectral dimension, hence the spectral features extracted from this axis are limited and constrained to a single Hadamard product, since there is no movement of filter in this domain. The kernel convolves over the spatial local neighborhood of the central pixel. Finally, the 3D variant uses three blocks, where each one incorporates two layers with 3D convolution. It extracts spectral-spatial features by sliding over all of three dimensions of the input cube. The output of every module is concluded with a regressor part that incorporates 3 consecutive fully-connected layers, where the number of last activations is equal to the number of endmembers in the image. For non-linearities we utilize the ReLU activation function.

We further extend the already defined architectures by introducing more architectural variations. First, we utilize a model with dimensionality reduction via two additional 3D convolutional layers without padding. We refer to it as MB-DR. Furthermore, based on such architecture, we add residual connections to accelerate the process of training and improve the gradient flow for deeper layers (MB-Res). Finally, keeping the last modifications, we extend the training scheme to incorporate two-step approach: *i*) train each branch separately and *ii*) fine-tune the entire model or only the regressor module. In *ii*), the first approach incorporates a pre-training the feature extractor, named MB-PT, whereas when training only the regressor part, thus this approach may be considered as transfer learning, we refer to it as MB-TL. In the experiments, we focus on three well-known HU benchmark datasets, i.e., Samson, Urban and Jasper Ridge. Furthermore, we sample 30 tests sets via Monte Carlo cross-validation strategy that remain constant when changing the training set size for each benchmark.

The experiments showed that 1D and 3D modules provide the best performance in terms of the quality of HU. When combining 2D and 3D branches there is no statistically significant improvement when compared to the 3D convolution

model. Such phenomenon could be because of the fact that additional 2D convolution does not introduce any new informative context nor features during the fusion step. The 3D convolutional layers alone provide all the required data to properly estimate the abundances of the input cube. Furthermore, we observe noticeable improvements in HU when spectral and spatial-spectral features are fused together. Finally, within the noise injection experiments, we showed that our proposed models are robust against various difficult acquisition conditions and provide stable results. The experiments denoted that our method allows to outperform other classic and DL method for HU.

Contribution of the PhD Candidate:

The PhD Candidate was responsible for all aspects of the multibranch convolutional neural network development for hyperspectral unmixing, including designing and implementing the complete multibranch architecture with spectral, spatial, and spectral-spatial feature fusion capabilities. The PhD Candidate developed all architectural variations, implemented the two-step training approach with pre-training and transfer learning schemes, conducted comprehensive experiments across benchmark datasets (Samson, Urban, Jasper Ridge), performed Monte Carlo cross-validation with 30 test sets, analyzed the impact of varying training set sizes and noise injection on model robustness, conducted ablation studies, collected and analyzed all experimental results, and contributed to the complete manuscript including visualizations and tables. The PhD Candidate also contributed to the literature review and experimental setup design in collaboration with co-authors.

- **Lukasz Tulczyjew**, Michal Przewozniczek, Renato Tinós, Agata M. Wijata, and Jakub Nalepa. "CANNIBAL Unveils the Hidden Gems: Hyperspectral Band Selection via Clustering of Weighted Variable Interaction Graphs." In Proceedings of the Genetic and Evolutionary Computation Conference, pp. 412-421. 2024 [240].
  CORE ranking: A.

  Feature (Band) Selection for (Not Only) Hyperspectral Data:

  This work explores the potential of HSI, which captures rich information across numerous spectral bands, yet poses significant challenges in terms of transfer, storage, and analysis due to its high spectral and spatial dimensionality. To address these issues, we propose **CANNIBAL**, a geneti**C** **A**lgorithm with li**N**kage lear**NI**ng with unsupervised clustering for hyperspectral **BA**nd se**L**ection. This novel band selection algorithm that leverages unsupervised clustering of inter-band dependencies using Variable Interaction Graphs (VIGs), which are generated as a byproduct of Genetic Algorithm with Linkage Learning (GAwLL) optimization. By capturing inter-band correlations, CANNIBAL efficiently identifies and selects the most informative bands, substantially reducing data dimensionality without compromising the quality of downstream tasks.

We validate CANNIBAL on two representative HSI use cases—hyperspectral unmixing and segmentation—demonstrating that it consistently outperforms existing band selection techniques in terms of performance and compression. Its flexible design supports both parametric and non-parametric configurations, making it adaptable to a wide range of Earth observation scenarios, particularly when only certain spectral bands are relevant to specific tasks.

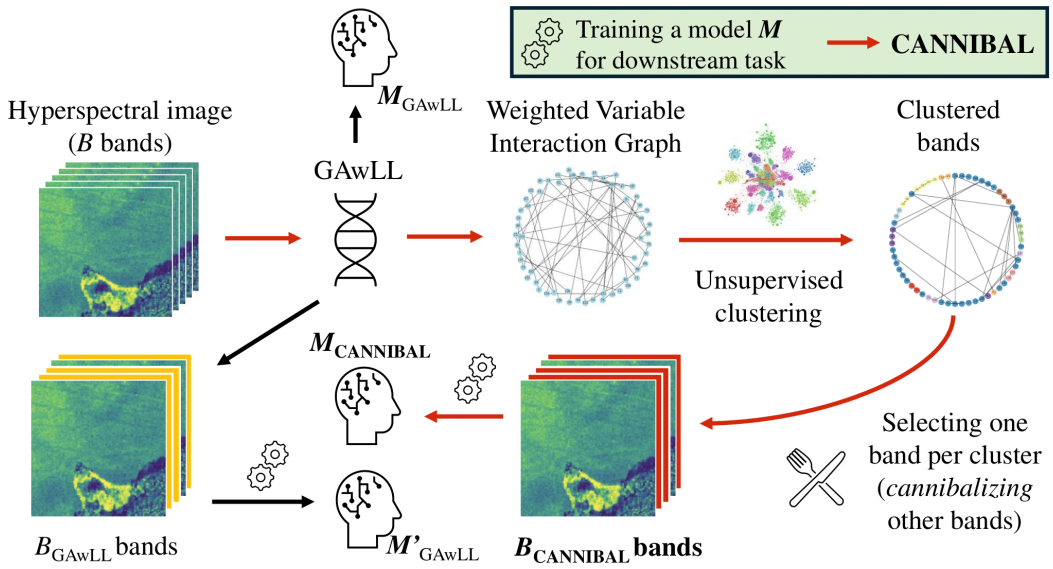


FIGURE 2.2: The flow diagram of proposed CANNIBAL method included in our work [240]. The start of the flow represents the HSI on the upper left part. Subsequently, the GAwLL [230] is employed to obtain the optimal solution with the inherently trained model M_{GAwLL} as well as the selected set of bands denoted as B_{GAwLL} bands. Those artifacts may be used directly, however, in our approach we utilize the side product of this optimization, i.e., the weighted VIG, it undergoes unsupervised clustering to detect the most related cliques of bands. Such groups denote features that are highly correlated to each other and convey similar information that may be redundant. Based on this assumption and hypothesis, we select one band per cluster of similar features, i.e., cannibalize all other bands within those groups. Afterwards, we obtain B CANNIBAL bands and output our selected features $B_{CANNIBAL}$ bands and the trained model $M_{CANNIBAL}$. Finally, there is another alternative of obtaining adjustable model based on bands generated by GAwLL algorithm, i.e., rejecting the embedded GAwLL model and training from scratch basing on the selected bands, this model is denoted as M'_{GAwLL}. This figure was also utilized in the PhD Candidate's work [240].

The overall architecture and operational flow of the proposed CANNIBAL method is illustrated in Figure 2.2. Starting from the input HSI, the GAwLL algorithm [230] is first used to perform optimization and obtain both a trained model M_{GAwLL} and a corresponding subset of selected bands B_{GAwLL}. While these outputs may be used directly, our method instead exploits the weighted Variable Interaction Graph (VIG) produced as a side product of GAwLL. This graph is subjected to unsupervised clustering to identify tightly connected

cliques of bands that are likely redundant due to high inter-correlation. Based on this structure, one representative band is selected from each cluster, effectively reducing dimensionality by cannibalizing redundant features. The final output consists of the refined subset of bands $B_{CANNIBAL}$ and the associated trained model $M_{CANNIBAL}$. Alternatively, one may discard the embedded GAwLL-trained model and retrain a new model M'_{GAwLL} from scratch using only the selected bands. This flexibility, highlighted in the diagram, underscores the modular and adaptable nature of the CANNIBAL framework.

A key novelty of our approach lies in the utilization of the side product of the GAwLL algorithm [230], namely the VIG, rather than relying solely on the final optimal solution to the underlying optimization problem. This strategic shift enables us to extract structural information about feature dependencies, offering a new perspective on the band selection process. Moreover, the CANNIBAL framework is designed to be highly modular: nearly every step in the processing pipeline can be replaced with an alternative algorithm that fulfills the same underlying function. For example, the clustering component is flexible and can incorporate various algorithms, both parametric and non-parametric, depending on the needs of the application or the desired level of control over the output band count. The cannibalization step itself, which selects a representative feature from each cluster, is also customizable; while our default implementation selects the feature with the highest total similarity, alternative statistical metrics such as intra-cluster variance or standard deviation can be used instead. Additionally, the GAwLL component can be substituted with any optimization technique capable of learning linkage structures or generating weighted interaction graphs as a side effect. We have evaluated CANNIBAL across multiple hyperspectral datasets, confirming its ability to generalize across different types of targets and endmembers. Finally, we provide an open-source implementation of CANNIBAL, publicly available at `https://github.com/V-o-y-a-g-e-r/CANNIBAL/`, to encourage further research and application in the hyperspectral imaging community.

The experiments can be decomposed into two parts: *i*) HU utilizing Urban benchmark dataset, incorporating six various endmembers and *ii*) hyperspectral segmentation task with Pavia University benchmark dataset with 9 targets. We compared our model with SOTA feature selection methods widely utilized in remote sensing imagery. From the unmixing experiments we conducted that CANNIBAL outperforms all baseline methods for most utilized metrics. Furthermore, for hyperspectral segmentation our method also outperforms other selected SOTA algorithms.

Contribution of the PhD Candidate:

The PhD Candidate was the primary architect responsible for designing and developing the complete CANNIBAL algorithm framework, including the novel

approach of utilizing VIGs as byproducts of GAwLL optimization for hyperspectral band selection. The PhD Candidate implemented the entire methodology including the unsupervised clustering of inter-band dependencies, the "cannibalization" process for selecting representative bands from clusters, and the modular framework design supporting both parametric and non-parametric configurations. The PhD Candidate conducted comprehensive experiments across hyperspectral unmixing (Urban dataset) and segmentation (Pavia University dataset) tasks, performed implementation and comparative analysis against SOTA feature selection methods, analyzed all experimental results, developed the open-source implementation available on GitHub, and contributed the complete manuscript including visualizations, flow diagrams, and tables. The PhD Candidate also contributed to the literature review and collaborated with co-authors on manuscript refinement.

2.2 On the Reproducibility of Machine Learning Research

Ensuring reproducibility is a cornerstone of scientific progress, especially in the context of ML, where methodological pitfalls such as data leakage, unreported randomness, or poor documentation can lead to overly optimistic and non-replicable results. As emphasized by Kapoor et al. [109], reproducibility crises in ML-based science are often driven by unnoticed data leakage, which can critically distort the validity of conclusions in fields ranging from political science to biology. Their study outlines a taxonomy of eight distinct types of leakage and finds evidence of reproducibility problems in at least 17 scientific fields, affecting nearly 300 publications. The consequences of such errors include inflated model performance and unjustified claims that newer ML techniques outperform traditional statistical models—claims which often collapse under rigorous reproduction attempts. These findings highlight the urgent need for better reporting standards and reproducibility practices in ML research, such as model info sheets proposed by the authors.

In light of this, we ensure full transparency and reproducibility in our work by publicly releasing all relevant codebases, datasets (where licensing permits), model architectures, and configurations. Each repository includes clear documentation, step-by-step reproduction instructions, and references to corresponding publications:

- **CANNIBAL: Band Selection via Clustering of Variable Interaction Graphs**
  https://github.com/V-o-y-a-g-e-r/CANNIBAL/
  Contains the full implementation of the CANNIBAL algorithm used for band selection in hyperspectral imaging, along with scripts for running experiments described in [240].

- Jakub Nalepa, Michal Myller, **Lukasz Tulczyjew**, and Michal Kawulok. "Deep ensembles for hyperspectral image data classification and unmixing." Remote

Sensing 13, no. 20 (2021): 4133 [178].
https://github.com/ESA-PhiLab/hypernet/tree/master/beetles
Includes deep learning models, experimental pipelines, and data processing scripts for hyperspectral unmixing under varying dataset sizes.

- **Lukasz Tulczyjew**, Michal Kawulok, Nicolas Longépé, Bertrand Le Saux, and Jakub Nalepa. "Unbiased Validation of Hyperspectral Unmixing Algorithms." In IGARSS 2023-2023 IEEE International Geoscience and Remote Sensing Symposium, pp. 7344-7347. IEEE, 2023 [235].
https://gitlab.com/jnalepa/hu_validation
Provides implementations of the training and testing pipelines, the proposed non-overlapping sampling method as well as model architectures.

- **Lukasz Tulczyjew** and Jakub Nalepa. "Investigating the impact of the training set size on deep learning-powered hyperspectral unmixing." In 2021 IEEE International Geoscience and Remote Sensing Symposium IGARSS, pp. 2024-2027. IEEE, 2021 [238].
https://github.com/ESA-PhiLab/hypernet/tree/master/beetles Includes implemented and evaluated models, training and evaluation pipelines. Allows for full experiments reproducibility.

- **Lukasz Tulczyjew**, Michal Kawulok, Nicolas Longépé, Bertrand Le Saux, and Jakub Nalepa. "A multibranch convolutional neural network for hyperspectral unmixing." IEEE Geoscience and Remote Sensing Letters 19 (2022): 1-5 [233].
https://gitlab.com/jnalepa/mbhu
Provides implementations of all of the multi-branch architectures as well as the literature baselines utilized in the study. We also include the required codebase to run all experiments coupled with implemented metrics.

All repositories are open-source and freely available, with experiments configured for reproducibility through fixed seeds, consistent preprocessing, and self-contained evaluation scripts. By adopting these best practices, we align with the growing standards in ML research aimed at increasing transparency, trust, and long-term impact.

2.3 Published Works Not Included in This Thesis

1. Pablo Ribalta Lorenzo, **Lukasz Tulczyjew**, Michal Marcinkiewicz, and Jakub Nalepa. "Hyperspectral band selection using attention-based convolutional neural networks." IEEE Access 8 (2020): 42384-42403 [150].

   Contribution of the PhD candidate:

   The PhD Candidate was responsible for research and development of the proposed method for feature, i.e., band selection. Furthermore, the PhD Candidate was executing all experiments and gathering results. Moreover, the PhD Candidate was involved in writing the manuscript.
   Impact factor: 3.367. Ministerial score (points): 100.

2. Jakub Nalepa, Michal Myller, Marcin Cwiek, Lukasz Zak, Tomasz Lakota, **Lukasz Tulczyjew**, and Michal Kawulok. "Towards on-board hyperspectral satellite image segmentation: Understanding robustness of deep learning through simulating acquisition conditions." Remote Sensing 13, no. 8 (2021): 1532 [172].

   Contribution of the PhD candidate:

   The PhD Candidate was responsible for conducting experiments and implementing the investigated methods and the overall required codebase.
   Impact factor: 5.349. Ministerial score (points): 100.

3. **Lukasz Tulczyjew**, Michal Kawulok, and Jakub Nalepa. "Unsupervised feature learning using recurrent neural nets for segmenting hyperspectral images." IEEE Geoscience and Remote Sensing Letters 18, no. 12 (2020): 2142-2146 [236].

   Contribution of the PhD candidate:

   The PhD Candidate's work encompassed implementing the proposed unsupervised method as well as other literature approaches, executing all experiments, statistical analysis and partial writing of the manuscript.
   Impact factor: 3.966. Ministerial score (points): 140.

4. Jakub Nalepa, Bertrand Le Saux, Nicolas Longépé, **Lukasz Tulczyjew**, Michal Myller, Michal Kawulok, Krzysztof Smykala, and Michal Gumiela. "The hyperview challenge: Estimating soil parameters from hyperspectral images." In 2022 IEEE International Conference on Image Processing (ICIP), pp. 4268-4272. IEEE, 2022 [171].

   Contribution of the PhD candidate:

   The PhD Candidate was responsible for technical part of the challenge, i.e., implementation, establishing the scoring metric, providing tutorial notebooks for competitors and was responsible for evaluating users' submissions to assess the reproducibility and correctness.
   CORE ranking: B.

5. **Lukasz Tulczyjew**, Michal Kawulok, Nicolas Longépé, Bertrand Le Saux, and Jakub Nalepa. "Graph neural networks extract high-resolution cultivated land maps from Sentinel-2 image series." IEEE Geoscience and Remote Sensing Letters 19 (2022): 1-5 [234].

   Contribution of the PhD candidate:

   The PhD Candidate implemented the proposed algorithm as well as baseline literature methods, performed all experiments and took part in construction of the scientifc article.
   Impact factor: 4.8. Ministerial score (points): 140.

6. **Lukasz Tulczyjew**, Michal Myller, Michal Kawulok, Daniel Kostrzewa, and Jakub Nalepa. "Predicting risk of satellite collisions using machine learning." Journal of Space Safety Engineering 8, no. 4 (2021): 339-344 [237].

   Contribution of the PhD candidate:

   The work of PhD Candidate was concerning implementation and evaluation of all of the utilized models and approaches for detecting satellite collisions. Furthermore, the PhD Candidate was the main contributor during writing of the manuscript.
   Impact factor: 1.0. Ministerial score (points): 20.

7. Jakub Nalepa, **Lukasz Tulczyjew**, Michal Myller, and Michal Kawulok. "Hyperspectral image classification using spectral-spatial convolutional neural networks." In IGARSS 2020-2020 IEEE International Geoscience and Remote Sensing Symposium, pp. 866-869. IEEE, 2020 [180].

   Contribution of the PhD candidate:

   The PhD Candidate was responsible for executing the greater majority of experiments, implementing the proposed approach as well as utilized augmentation methods. Moreover, the PhD Candidate collected all results and created tables in the manuscript.
   CORE Ranking: C.

8. Tomasz Jastrzab, Michal Myller, **Lukasz Tulczyjew**, Miroslaw Blocho, Michal Kawulok, Adam Czornik, and Jakub Nalepa. "Standardized validation of vehicle routing algorithms." Applied Intelligence 54, no. 2 (2024): 1335-1364 [103].

   Contribution of the PhD candidate:

   In this work, the PhD Candidate was responsible for implementing the Benchmark Generator that was utilized in the study for validation of functional and non-functional abilities of the routing problem solvers. Furthermore, the PhD Candidate took part in implementing a set of quality metrics that help to quantify various aspects of solutions, such as their profitability.
   Impact factor: 3.4. Ministerial score (points): 70.

9. Tomasz Tarasiewicz, **Lukasz Tulczyjew**, Michal Myller, Michal Kawulok, Nicolas Longépé, and Jakub Nalepa. "Extracting high-resolution cultivated land maps from sentinel-2 image series." In IGARSS 2022-2022 IEEE International Geoscience and Remote Sensing Symposium, pp. 175-178. IEEE, 2022 [226].

   Contribution of the PhD candidate:

   The PhD Candidate took part in the ideation and implementation of the proposed method for extracting high-resolution cultivated land maps from Sentinel-2 image series.
   CORE Ranking: C.

10. Tomasz Jastrzab, Michal Myller, **Lukasz Tulczyjew**, Miroslaw Blocho, Wojciech Ryczko, Michal Kawulok, and Jakub Nalepa. "Particle swarm optimization configures the route minimization algorithm." In International Conference on Computational Science, pp. 80-87. Cham: Springer International Publishing, 2022 [104].

    Contribution of the PhD candidate:

    The PhD Candidate was involved in implementing the proposed method for route minimization as well as executing all experiments presented in the manuscript.
    CORE Ranking: A.

11. **Lukasz Tulczyjew**, Kinan Jarrah, Charles Abondo, Dina Bennett, and Nathanael Weill. "LLMcap: Large Language Model for Unsupervised PCAP Failure Detection." In 2024 IEEE International Conference on Communications Workshops (ICC Workshops), pp. 1559-1565. IEEE, 2024 [232].

    Contribution of the PhD candidate:

    The PhD Candidate proposed and implemented the Large Language Model-based method introduced in the manuscript. Furthermore, the PhD Candidate was responsible for executing all experiments, statistical analysis and writing of the scientific article.
    CORE ranking: Co-located with ICC rank—B.

12. Vladimir Zaigrajew, Hubert Baniecki, **Lukasz Tulczyjew**, Agata M. Wijata, Jakub Nalepa, Nicolas Longépé, and Przemyslaw Biecek. "Red teaming models for hyperspectral image analysis using explainable AI.". In ICLR 2024 Machine Learning for Remote Sensing (ML4RS) Workshop (2024) [270].

    Contribution of the PhD candidate:

    The PhD Candidate was responsible for providing and training the required top-ranked models for explanatory model analysis. Based on those models, the read teaming methodology for hyperspectral image analysis was conducted.
    CORE ranking: Co-located with ICLR rank—A*.

13. **Lukasz Tulczyjew**, Ihor Biruk, Murat Bilgic, Charles Abondo, and Nathanael Weill. "PCAPVision: PCAP-Based High-Velocity and Large-Volume Network Failure Detection." In Proceedings of the 2024 SIGCOMM Workshop on Networks for AI Computing, pp. 26-33. 2024 [231].

    Contribution of the PhD candidate:

    The PhD Candidate proposed the methodology for Packet Capture failure detection and was responsible for implementing and conducting all experiments presented in the manuscript. Furthermore, the PhD Candidate was involved in writing of the scientific article.
    CORE ranking: Co-located with ACM SIGCOMM rank—A*.

14. Bartosz Grabowski, Agata M. Wijata, **Lukasz Tulczyjew**, Bertrand Le Saux, and Jakub Nalepa. "Soil Analysis with Very Few Labels Using Semi-Supervised Hyperspectral Image Classification." In IGARSS 2024-2024 IEEE International Geoscience and Remote Sensing Symposium, pp. 407-411. IEEE, 2024 [80].

    Contribution of the PhD candidate:

    The PhD Candidate took part in ideation and implementation of semi-supervised learning pipeline for elaborating deep learning models for multi-class hyperspectral image classification.
    CORE Ranking: C.

15. Agata M. Wijata, Tomasz Lakota, Marcin Cwiek, Bogdan Ruszczak, Michal Gumiela, **Lukasz Tulczyjew**, Andrzej Bartoszek, Nicolas Longépé, Krzysztof Smykala, and Jakub Nalepa. "Intuition-1: Toward In-Orbit Bare Soil Detection Using Spectral Vegetation Indices". In IGARSS 2024-2024 IEEE International Geoscience and Remote Sensing Symposium, pp. 1708-1712. IEEE, 2024 [253].

    Contribution of the PhD candidate:

    The PhD Candidate was responsible for preparing the dataset used for benchmarking and experiments as well as implementing the employed quality metrics.
    CORE Ranking: C.

16. Wiktor Gacek, **Lukasz Tulczyjew**, Agata M. Wijata, Nicolas Longépé, Bertrand Le Saux, and Jakub Nalepa. "Estimating Soil Parameters from Hyperspectral Images using Ensembles of Classic and Deep Machine Learning Models." In IGARSS 2024-2024 IEEE International Geoscience and Remote Sensing Symposium, pp. 412-416. IEEE, 2024 [67].

    Contribution of the PhD candidate:

    The PhD Candidate was responsible for preparing the dataset used in experiments, implementing metrics for evaluation, and helping during the experimental phase.
    CORE Ranking: C.

17. Jakub Nalepa, **Lukasz Tulczyjew**, Bertrand Le Saux, Nicolas Longépé, Bogdan Ruszczak, Agata M. Wijata, Krzysztof Smykala, Michal Myller, Michal Kawulok, Pidvan Salih Kuzu, Frauke Albercht, Caroline Arnold, Mohammad Alasawedah, Suzanne Angeli, Delphine Nobileau, Achille Ballabeni, Alessandro Lotti, Alfredo Locarini, Dario Modenni, Paolo Tortora, Michal Gumiela. "Estimating Soil Parameters From Hyperspectral Images: A benchmark dataset and the outcome of the HYPERVIEW challenge." IEEE Geoscience and Remote Sensing Magazine (2024) [179].

    Contribution of the PhD candidate:

    The PhD Candidate was responsible for preparing the benchmark dataset, implementing evaluation metrics, implementing the validation procedure, grading all of the competitors submissions, providing experimental results, reproducing top-scoring methods and took part in writing of the manuscript.
    Impact factor: 10.283. Ministerial score (points): 140.

18. Bernard Valentin, Leslie Gale, Jannes Lathouwers, Garin Smith, Marco Cerri, Marco Pellegrini, Francesco Ferrante, Jakub Nalepa, **Lukasz Tulczyjew**, Jakub Sadel, Mateusz Przeliorz, Daniel Kostrzewa, Vaclav Svaton, "AIOPEN–Platform Extensions With AI Capabilities". Proceedings of the 2023 conference on Big Data from Space (BiDS'2023) [242].

    Contribution of the PhD candidate:

    The PhD Candidate took part in planning and implementation of the forest cover monitoring system as one of the services offered by the proposed platform.

Chapter 3

Background: Deep Artificial Neural Networks

In this chapter, we describe the foundation of Deep Artificial Neural Networks (DNNs). The first part encapsulates the mathematical formulation of the pivotal concepts and data structures. In the subsequent subsections, we introduce more advanced modifications of standard DNNs like CNNs and Graph Neural Networks (GNNs).

3.1 Forward Propagation

A standard DNN operates as a composite function composed of multiple layers, each performing a transformation on its input to eventually produce a prediction. In the forward propagation process, input data—represented as *samples*—are passed through the network layer by layer. Each sample is described by a set of *features*, which constitute the dimensions of the input space and serve as independent variables in the model. These features are problem-specific and can vary significantly in number, depending on the domain and task complexity.

However, as the number of input features increases, the model may struggle to generalize effectively due to the so-called *curse of dimensionality* [6]. This phenomenon describes the exponential growth in computational complexity and data sparsity as dimensionality rises, often leading to overfitting and degraded performance. In such scenarios, the forward propagation process becomes less efficient, as the network must process a high-dimensional input space, much of which may contain redundant or irrelevant information.

To mitigate these issues and improve learning efficiency, feature selection becomes crucial. Feature selection aims to reduce the dimensionality of the input space by identifying and retaining only the most informative variables. This not only enhances model interpretability and training speed but also improves generalization. In the literature, three main categories of feature selection approaches are commonly distinguished: (*i*) *wrapper methods*, which evaluate subsets of features based on model performance; (*ii*) *filter methods*, which rely on statistical properties such as correlation or mutual information; and (*iii*) *embedded methods*, which integrate feature selection directly into the model training process [28], [108], [203]. By applying such

techniques, the model can focus its forward computations on the most relevant data dimensions, thereby reducing the burden of dimensionality and enhancing predictive capability.

Similarly to the input data, predicted output is also problem-specific and can be composed of any number of dimensions. It can also vary between continuous or discrete values, coupled with necessary post-processing, e.g., thresholding.

The entire mechanism can be mathematically described as matrix multiplication, where the input vector (for a single sample) or matrix (for an entire batch) is combined with the weight matrix. The operation can be formulated as:

$$\mathbf{Z}^{(l)} = \mathbf{X}^{(l-1)}\mathbf{W}^{(l)} + \mathbf{b}^{(l)}, \tag{3.1}$$

where:

- $\mathbf{X}^{(l-1)}$ is the input matrix to layer l, with shape (n, d_{l-1}), where n is the batch size and d_{l-1} is the number of features from the previous layer,
- $\mathbf{W}^{(l)}$ is the weight matrix of layer l with shape (d_{l-1}, d_l),
- $\mathbf{b}^{(l)}$ is the bias vector of layer l with shape $(1, d_l)$, and
- $\mathbf{Z}^{(l)}$ is the pre-activation output of layer l.

3.2 Non-Linearity

The DNNs feed forward mechanism processes an input sample to obtain the model prediction. Such mechanism, without any mathematical modifications defined in Subsection 3.1, incorporates only linear operations thus is unable to model non-linear relationships that can be inherently present in the data. To overcome this issue and enable more complex data modeling, DNNs incorporate activation functions. Those structures are utilized in both, forward and backward propagation. Without activation functions, the model would only perform linear transformations of the input, hence rendering itself with limited capability.

The formula with added activation function can be presented as:

$$\mathbf{A}^{(l)} = \sigma(\mathbf{Z}^{(l)}), \tag{3.2}$$

where $\sigma(\cdot)$ denotes an element-wise non-linear activation function (e.g., ReLU, sigmoid, tanh, see Subsection 3.2 for detailed formulas), and $\mathbf{A}^{(l)}$ is the output (activation) of layer l, which becomes the input for the next layer.

With the unprecedented development of DL, the number of different activation functions is expanding rapidly. Each variation has its own advantages and use cases. Below, we present the set of activation functions used across the models introduced in this work. In addition to standard choices such as ReLU, Sigmoid, or Tanh,

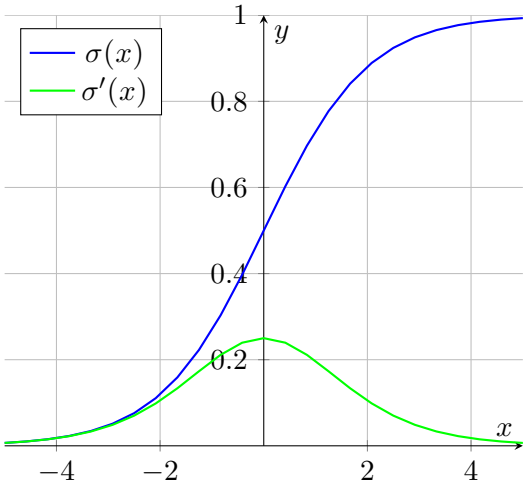


FIGURE 3.1: Sigmoid activation function (blue) and its derivative (green) utilized during backward propagation.

we also employ task-specific or less common activation functions where appropriate, depending on the architecture and optimization needs of each model.

The pivotal and one of the most known activation functions is called the *sigmoid* function, it is depicted in Diagram 3.1. It can be characterized by a distinctive *S*-shaped curve, which centers directly on 0.5 in its domain. The codomain spans from 0 to 1 both inclusive ranges. This characteristic allows to control the output values and prevent from exploding gradient problem [14], [192].

The sigmoid activation function can be described as:

$$\sigma(x) = \frac{1}{1 + e^{-x}}, \tag{3.3}$$

where x is the input and e denotes the exponential function.

Another activation function that is widely utilized is *hyperbolic tangent* (tanh). The output range is defined from -1 to 1, both inclusive. Its shape is presented in Diagram 3.2.

The tanh nonlinearity can be mathematically formulated as:

$$\tanh(x) = \frac{e^{2x} - 1}{e^{2x} + 1}. \tag{3.4}$$

The tanh activation function is most commonly utilized in Recurrent Neural Networks (RNNs) [162]–[165], [266] and their successors with more advanced gating mechanisms, such as GRUs and LSTMs [40], [91]. This is primarily due to its ability to center the activations around zero, producing outputs in the range $[-1, 1]$. Unlike the sigmoid function, which outputs values in $[0, 1]$, tanh enables more balanced gradients during training, reducing the risk of biased updates and improving convergence. In RNNs, where information is propagated across many time steps, tanh helps maintain

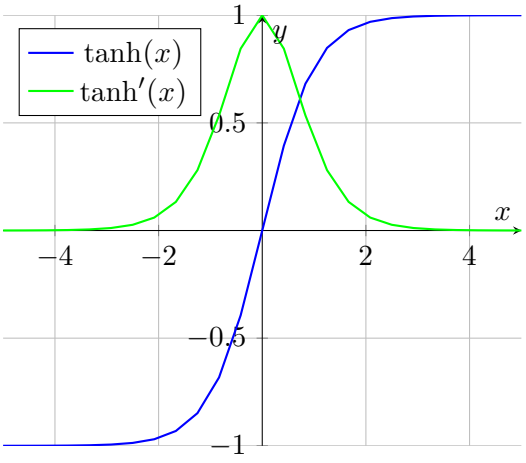


FIGURE 3.2: Tanh non-linearity (blue) with its derivative (green).

signal strength over time, making it especially suitable for handling temporal dependencies in sequences. Furthermore, both GRU and LSTM architectures explicitly incorporate tanh in their update and output gates, leveraging its smooth non-linearity to regulate memory cell values and improve learning dynamics in long-range sequence modeling.

With the recent advancements in DL, most modern models have moved away from traditional activation functions such as Sigmoid or Tanh, in favor of novel SOTA alternatives. One of the main reasons behind this shift is the need to mitigate the vanishing gradient problem [90]. This phenomenon occurs during backpropagation when gradients become progressively smaller as they are propagated through multiple layers, eventually approaching zero. Such behavior hinders effective learning and slows convergence, especially in deep architectures. For example, the derivative of the sigmoid function (see Figure 3.1) quickly saturates, reducing the gradient signal to near-zero values.

Another related challenge is the exploding gradient problem, which arises when large weight values in the forward pass cause activations and gradients to grow exponentially. This results in instability during training, making it difficult for the optimization process to converge to meaningful local minima. These two issues — vanishing and exploding gradients — both stem from the compounding effect of layer-by-layer transformations in deep networks. Modern activation functions and normalization techniques have been introduced to alleviate these issues, enabling deeper and more stable model training.

The recent advancements lead to creating different modifications of Rectified Linear Unit (ReLU) function [170]. The ReLU nonlinearity can be formulated as:

$$\text{ReLU(x)} = \begin{cases} x, & \text{if } x > 0 \\ 0, & \text{otherwise.} \end{cases} \tag{3.5}$$

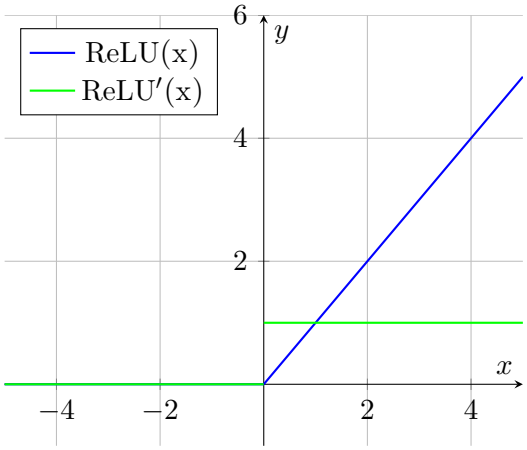


FIGURE 3.3: ReLU activation function and its derivative marked in blue and green colors.

This activation function owes its popularity by numerous reasons, one of them being its simplicity and adaptability. In Figure 3.3, it is visible that the output gradient denoted in green is constant and allows for smooth training by mitigating the exploding as well as vanishing gradient problems [14], [90], [192]. However, it is important to notice that when the output values are below or equal to 0, the gradient becomes 0 as well. In such situations, the training is impossible because the weight updates are non-existing. This problem is referred to in literature as *dying* ReLU [155]. To overcome this issue, there are numerous modifications of this pivotal activation function that allow for effective parameter updates, even when the output activations are negative. There are worth mentioning variations of this function such as: Leaky ReLU [260], Exponential Linear Unit [44], Parametric leaky version of a Rectified Linear Uni (PReLU) [200], or Scaled Exponential Linear Unit (SELU) [200]. Each of the mentioned non-linearities aim to improve ReLU's stability during training, mitigate the mentioned issue for negative outputs, enhance smoothness of its derivative, and adapt to a particular use case.

3.3 Backward Propagation

The backward propagation process is the foundation of training mechanism that allows for adjusting the model to a particular task. It incorporates the process of *gradient descent* [206]. Its main goal is to minimize the error calculated by comparing the model output values with the expected targets, often referred to as *ground-truth* (GT). The error can be computed using various functions, depending on the range of values. It is worth emphasizing that there can be millions of trainable parameters inside the model, thus the search space is highly complex and sensitive to tiny changes in any dimension. Furthermore, one parameter is equal to one dimension in the optimization space. Consecutively, weights are correlated to each other, meaning that

changing one weight can affect the others, due to the way the gradient is computed and applied during gradient descent process.

When the error is calculated in the output layer, the obtained partial derivatives are used in the previous layers, continuing to the first one. In the gradient descent method, the model is updated by subtracting a fraction of its magnitude against the current weight values. In Subsection 3.4, we describe the most known and used optimizers.

An *optimizer* in deep neural networks is an algorithm responsible for updating the weights of the model based on the computed gradients during backpropagation. Its goal is to minimize the loss function by iteratively adjusting the model parameters in the direction that reduces prediction error. Optimizers determine how large the update step should be (via the learning rate), how momentum or history of gradients should influence updates, and how adaptive mechanisms should adjust learning dynamically. While the basic *Gradient Descent* method provides a foundational framework, in practice, more advanced optimizers such as *Adam*, *RMSprop*, or *Adagrad* are preferred due to their improved convergence speed, ability to escape local minima, and robustness to sparse gradients [54], [116], [271]. These optimizers are key to efficiently training deep networks, particularly in high-dimensional and non-convex problem spaces.

3.4 Optimizers

Following an overview of the most commonly used and fundamental activation functions and their significance in deep learning, in this section we focus on exploring the core optimization mechanics of DNNs in greater detail. As previously mentioned, a standard DNN can be viewed as a complex, composite function where each stage involves two operations: matrix multiplication and addition, followed by a nonlinear activation. Each stage corresponds to a specific layer in the network, each containing its own set of parameters, typically referred to as weights. These weights are fine-tuned during training using the backpropagation algorithm. This method is essential for supervised learning and is based on the gradient descent optimization technique.

The most classic and known gradient descent method is referred to as stochastic gradient descent (SGD). Because of the "stochastic" prefix, its main function is to calculate and apply the gradient over a very small batch or usually a single sample. Such phenomenon renders it as unstable and versatile based on the correctness of the model over such small batch. The scaling factor which minimizes the effect of full gradient magnitude is often referred to as η. This hyper-parameter is global, i.e., affects all of the learnable parameters, hence is difficult to appropriately choose.

Another noteworthy optimizer is *SGD with momentum* [24]. The core concept of this approach involves incorporating the gradients accumulated from previous iterations when calculating the update for the current step. If the current gradient aligns with the direction of the preceding gradients, meaning they share the same sign,

the update step is amplified. Conversely, when the gradients have opposite signs, momentum decreases the size of each update. This mechanism not only accelerates the training process but also reduces fluctuations within the optimization landscape, thereby preventing the overshooting of optimal points.

A variant of the previously mentioned method is known as the *Nesterov accelerated gradient* [24], representing another significant advancement in optimization techniques. The fundamental distinction between this approach and the standard momentum method is that instead of computing the gradient at the current position, the optimizer takes an additional step in the direction of the accumulated gradient, referred to as the velocity, before evaluating the gradient. This implies that the gradient accumulated from prior iterations is used twice in determining the update step. This strategy enables the optimizer to effectively preview the next step in the direction of the velocity, manage the gradient's magnitude, and prevent overshooting the minima.

The current growing preference for deep learning-based end-to-end solutions that do not rely on any *a priori* task-specific knowledge stems from exponentially increasing data growth rate [223]. Unsurprisingly, these methods are favored over manual approaches due to their greater convenience and quicker implementation. Nonetheless, the swift expansion of data comes with several drawbacks. A significant challenge is the existence of a large number of features that are difficult to interpret and cause the previously mentioned curse of dimensionality [6]. When combined with frequent data sampling, this leads to numerous redundant features where values either show minimal variation or remain consistently at zero. This situation results in *sparse* datasets, characterized by many features predominantly containing zeros. Such sparsity complicates the optimization of DNNs, as certain parameters are scarcely utilized during the feed-forward phase. Consequently, the scaled-down gradients calculated for these weights during the backward pass become almost negligible. To address this issue, an adaptive gradient-based optimization algorithm known as *Adagrad* was introduced in [54]. As previously discussed, Adagrad's primary role is to adjust the learning rate hyperparameter dynamically. It does so by reducing the update magnitude for frequently used features, thereby enforcing smaller updates. Conversely, for less common weights—resulting from input feature sparsity—it increases the gradient, making updates more substantial for those parameters.

Adadelta is another noteworthy gradient descent optimizer for DNNs, first presented in [271]. Its core innovation lies in addressing the key shortcoming of the Adagrad method: Adagrad tends to make the learning rate shrink drastically over time, owing to the continuous accumulation of squared gradients throughout training. In contrast, Adadelta cleverly limits the influence of historical gradients by maintaining an exponentially decaying average of squared gradients instead of storing their entire sum. By doing so, it effectively "forgets" older updates at a controlled pace, preventing the learning rate from diminishing too much as training progresses.

Beyond merely counteracting the steadily declining updates in Adagrad, Adadelta

also enhances stability when training models on complex tasks, such as those involving deep or recurrent network architectures. By striking a balance between adapting to recent gradients and not overemphasizing their long-term accumulation, this optimization strategy can help models converge more reliably and tune their parameters effectively across varying scales of features.

Beyond Adadelta, the RMSProp algorithm also tackles the challenge of a shrinking learning rate inherent in Adagrad by regulating the sum of squared gradients in a manner similar to Adadelta [206]. This approach ensures that gradients do not diminish as training progresses, leading to more consistent updates for DNN parameters.

On top of these advancements, the Adaptive Moment Estimation (Adam) optimizer introduced in [116] combines helpful ideas from both Adadelta and RMSProp to counteract the issue of diminishing learning rates. In addition, Adam maintains a memory of past gradients in the form of momentum, allowing it to adapt more efficiently to the curvature of the loss landscape. This momentum-like aspect is particularly beneficial near saddle points, where gradients can be weak or oscillatory. By keeping an exponentially decaying average of past gradients and their squares, Adam can accelerate convergence while preserving the stability that comes from controlling learning rates, making it one of the most widely used optimization methods in DL [83].

3.5 Convolutional Neural Networks

In the broader field of DL, convolution is a foundational mathematical operation at the heart of CNNs [129]. Similar to traditional fully-connected neural architectures, CNNs employ sets of learnable parameters to propagate inputs forward. However, instead of relying on a single, full-scale matrix multiplication, CNNs make use of smaller matrices, typically referred to as convolutional kernels or filters. These filters are systematically shifted across the input activation maps, which differs from how parameters are applied in standard matrix-to-matrix multiplication.

Each layered convolution can involve one or more of these filters, and the number of filters is determined by a hyperparameter defining the model's capacity to recognize different features. The output generated by each filter is known as a channel or a feature map. Mathematically, this "sliding" operation is usually performed by taking an element-wise (Hadamard) product of the filter with a portion of the input, followed by summing the product's values to generate a single response in the feature map.

Importantly, the shape of the convolutional kernel is tied to the nature of the input data. For 1D input signals (such as audio waveforms or other time series), the kernel spans all input channels but shifts along a single spatial or temporal axis. In contrast, image-based tasks rely on 2D filters, which move in both height and width dimensions of the image, while 3D filters can be applied to volumetric data or videos, shifting along three axes. This flexibility, combined with localized weight

sharing, allows CNNs to capture and learn hierarchical patterns in the data far more efficiently than standard fully-connected layers.

As an example, the 1D convolution operation can be mathematically formulated as:

$$o_z^{l,f} = \zeta(b^{l,f} + \sum_{c=1}^{C}\sum_{d=1}^{D} \theta_{c,d}^{l,f} \cdot i_{c,z+d}^{l-1}), \tag{3.6}$$

where i and o characters denote the input and output values within specific locations in the matrices. Additionally, θ and b demonstrate the adjustable weights, i.e., the filter and bias weights. Consequently, the C symbol shows the number of input feature maps and c points at a specific channel. The size of the filter is denoted by D. Finally, the l and f characters allow us to index the layer and its corresponding kernel, whereas ζ denotes the non-linearity function. This formula can be extended by adding more dimensions and allows to create 2- and 3D convolutional layers. In the Figure 3.4, we provide a visualization of a single convolutional layer and demonstrate how spatial and spectral features are aggregated in the subsequent layer, i.e., output activation map.

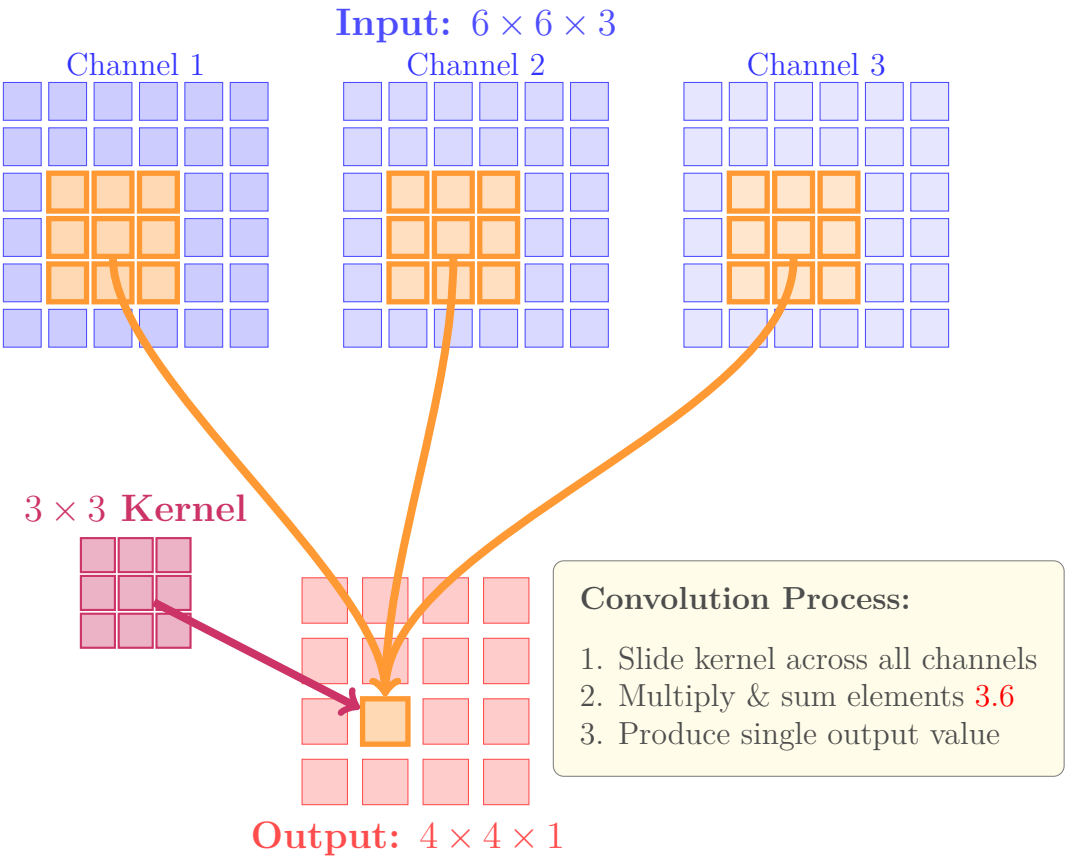


FIGURE 3.4: Simplified CNN showing how a 3×3 kernel convolves across $6 \times 6 \times 3$ input to produce a single 4×4 feature map. Input channels are colored in blue, whereas the output feature map is depicted in red. Finally, we highlight a single convolution operation in orange.

The size of each kernel's coverage along its dimensions can be adjusted as a hyperparameter, typically chosen based on the specific requirements of the problem at hand. This region that the kernel covers is often referred to as the *receptive field*. Beyond determining the kernel size, each convolutional layer also comes with additional hyperparameters that can significantly influence performance.

For instance, the stride specifies how many steps the filter "jump" as it slides across the input. A stride set to 1 means the kernel shifts one unit or pixel at a time, creating closely spaced convolution outputs. Increasing the stride reduces the total number of convolution operations, which in turn decreases the dimensions of the output feature maps.

Another key hyperparameter is padding. By appending extra values around the edges of the input data, one can preserve the same output size even when using larger strides or larger kernels. Although zero-padding is the most common (often referred to as "same" padding in some frameworks), other methods like adding mean, median, or symmetric padding are possible [86], [139].

Additionally, the kernel's layout itself can be modified through dilation, giving rise to dilated (or atrous) convolutions [139], [269]. In this setup, the elements of the kernel are spaced out, allowing the network to capture longer-range relationships without increasing the total number of learnable parameters. This approach can be especially useful in tasks like semantic segmentation or time-series analysis, where it is advantageous to expand the effective receptive field without overly inflating computational costs [140].

In most deep learning architectures, convolutional layers are often paired with pooling layers [210]. The principal reason for including pooling is to further reduce the dimensions of the output feature maps, thereby introducing a regularizing effect and speeding up the training, especially when dealing with large, high-resolution inputs. While pooling layers operate with a window-sliding mechanism similar to that of convolution, they differ in a key respect: instead of performing element-wise multiplications followed by a sum, a pooling layer calculates statistical measurements based on the values enclosed by its spatial window [76]. The most widely used types include *max pooling*, which selects the maximum value within the window, and *average pooling*, which computes the mean of the values. These operations help reduce spatial dimensions and control overfitting by providing a form of translation invariance. A related variant, known as global pooling, takes this a step further by aggregating the maximum or average over the entire spatial dimension, preserving only the channel information and thus reducing the dimensionality more drastically.

Another notable benefit of pooling layers is their contribution to spatial translation invariance [210]. By lowering the resolution of intermediate representations, small shifts or minor changes in the input data have less impact on the network's output. This is particularly advantageous for classification tasks involving high-quality images, where the precise position of a feature should not significantly alter the final prediction [126].

3.6 Recurrent Neural Networks

Recurrent Neural Networks (RNNs) are particularly effective whenever the data involves sequential or time-dependent patterns, such as in natural language processing (NLP) [162]–[164], [266]. Unlike conventional deep neural networks, RNNs incorporate an internal short-term memory mechanism that retains information from prior time steps, thus introducing an awareness of historical context. An illustrative application is sentiment analysis [214], where each sentence must be assigned a specific label reflecting its overall attitude. The key challenge lies in capturing the significance of the entire sequence of words, which standard DNNs often struggle to manage due to their lack of contextual memory. As demonstrated in numerous industries—ranging from consumer feedback and investment forecasting to security details [110]—effectively performing sentiment analysis can provide valuable insights that inform strategic decisions. Another important application of RNNs is in the analysis of satellite data, particularly for processing spatial-temporal sequences [277]. Satellite imagery and remote sensing data are often collected at regular intervals, forming time series that reflect dynamic changes in land cover, vegetation health, urban growth, or environmental conditions. RNNs, along with their gated extensions such as Long Short-Term Memory (LSTM) [91], [92] and Gated Recurrent Units (GRU) [40], [41], are especially well-suited for modeling such temporal dependencies due to their internal memory structure. For instance, in precision agriculture, those architectural modifications can be used to monitor crop growth patterns and predict crop semantic segmentation by analyzing multispectral data over time, as proposed in the PhD Candidate's work in [236]. Similarly, in climate monitoring, they can help detect gradual environmental changes or extreme weather events [32]. These capabilities make RNNs a powerful tool for Earth observation and environmental management tasks, where understanding temporal context is essential. Each of these models adds gating components to the recurrent cell structure, maintaining more relevant information for longer sequences and mitigating both vanishing and exploding gradients. By better preserving historical data while regulating step-to-step updates, these architectures offer improved performance for tasks involving extended temporal or sequential inputs. In our work outlined in [236], we developed an unsupervised feature extraction technique based on an autoencoder, where the encoder operates through an RNN. The architecture is depicted in Figure 3.5. The embedded representation produced by this encoder is then transformed back into the original domain via a decoder constructed from multiple fully connected layers. Once training is finalized, the decoder is removed, and the compressed embeddings are passed to any desired clustering method. Experimental outcomes highlight that this approach achieves high-quality segmentation even without any ground-truth labels, frequently exceeding or matching the performance of state-of-the-art solutions, such as the 3D convolutional autoencoder model proposed in [175].

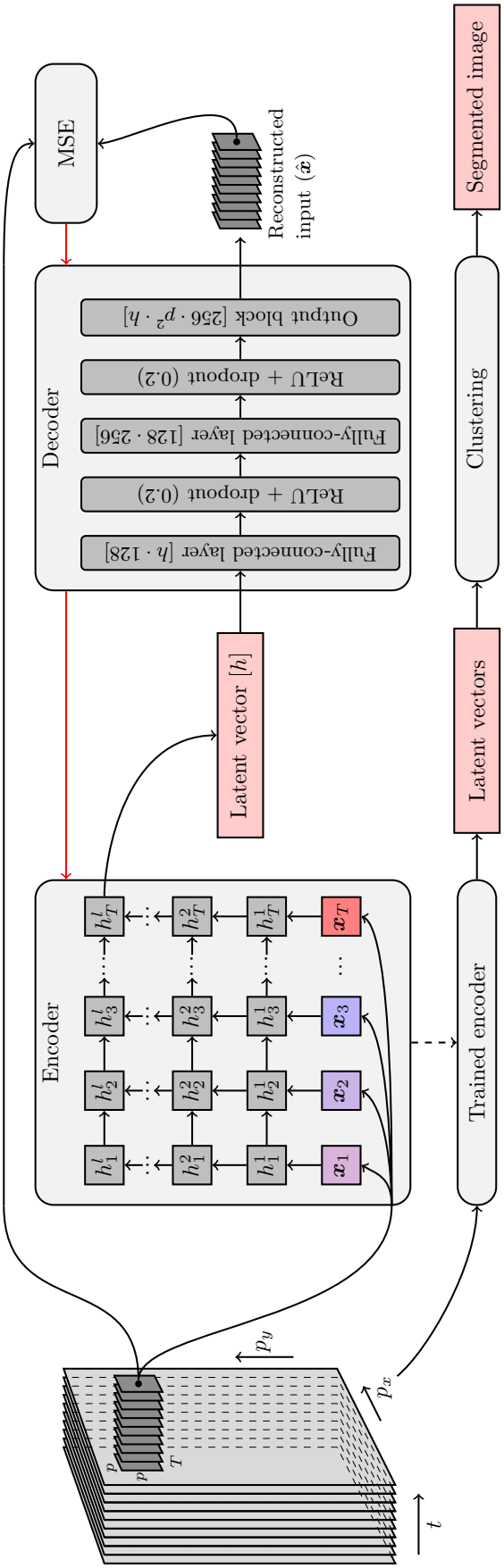


FIGURE 3.5: Flowchart of our proposed pipeline (we present the tensor dimensions in brackets). The red arrows show the error backpropagation (mean square error). The encoder is composed of 3-layer RNN. The decoder consists of fully-connected layers. This Figure was also utilized in the PhD Candidate's work in [236].

Standard RNNs frequently encounter considerable optimization challenges related to vanishing and exploding gradients, as documented in [14], [192]. These complications emerge when modeling extended temporal dependencies, causing gradients to either shrink toward zero or magnify excessively over successive time steps. A number of mitigation strategies have been explored, including well-known options such as L1 and L2 regularization [78]. In general machine learning settings, these techniques limit model overfitting by pushing certain parameters partially or entirely to zero, for L2 and L1 regularizers, respectively [184]. This also facilitates implicit feature selection, as weights that are driven toward zero effectively deactivate their associated features. In the deep learning context, you can incorporate L1 and L2 regularization as an extra penalty term in the loss function, which helps control parameter values and counteracts sudden gradient spikes that occasionally lead to overfitting [78], [185]. Another notable approach to addressing gradient explosion is gradient clipping [273], where any gradients that surpass a predefined threshold are scaled back, preventing large, destabilizing updates. Furthermore, it was experimentally proven in [273] that employing gradient clipping accelerates training of DNNs.

3.7 Graph Neural Networks

GNNs have emerged as a powerful class of deep learning models that operate on graph-structured data [258]. Unlike CNNs, which excel at processing grid-like arrangements of pixels, or RNNs, which are adept at handling sequential dependencies, GNNs are specifically designed to ingest and analyze sets of nodes and their corresponding edges, incorporating and allowing irregular and non-grid structures. By taking relationships among entities into account, GNNs provide a means to capture both local and global structural information in a single framework. This characteristic has proven invaluable for tasks such as node classification [259], link prediction [275], and graph-level classification in social networks [58], biological systems [282], chemistry [202], e-commerce [146], and many other fields [106], [258], [261], [286].

Graph data is noteworthy for its non-Euclidean structure, which sets it apart from more conventional domains like images or sequences. In a two-dimensional grid—such as an image—each pixel has a fixed arrangement with neighbors immediately above, below, left, or right, and in a time series every element follows a strict order. By contrast, graphs often exhibit an irregular connectivity pattern, where individual nodes can have a varying number of adjacent nodes. Simply reshaping these relationships into a flat format risks losing critical connections. To address this issue, GNNs employ message-passing algorithms [77] and specialized modules (often referred to as aggregators or updaters) that gather and refine information from each node's neighborhood in a principled manner [63].

From a functional standpoint, a GNN layer usually carries out three core steps [258]. First, it collects input signals—often called messages—from neighboring nodes. Next,

these messages are combined or aggregated to produce a single composite representation. Finally, this representation is used to update the node's internal embedding. By stacking multiple GNN layers, nodes gradually incorporate more contextual information from various parts of the graph, even those that are distant via multiple edges. This iterative exchange of data helps each node encode increasingly comprehensive insights into the broader topology [145].

The development of GNNs is fundamentally linked to the emergence of graph convolution techniques [276], which originate from concepts in spectral graph theory [43]. Early GNNs were implemented using spectral theory, where spectral graph theory studies fundamental graph properties using algebraic methods to analyze the spectrum of graphs [36], [38]. Early approaches utilized the eigen-decomposition of the graph Laplacian to define convolution operations in the frequency domain, enabling the generalization of traditional convolutional operations to non-Euclidean graph-structured data. A notable example is the work by Kipf and Welling [15], who presented a scalable approach for semi-supervised learning on graph-structured data based on an efficient variant of convolutional neural networks which operate directly on graphs, motivated by a localized first-order approximation of spectral graph convolutions [117]. While spectral-based techniques yielded strong performance in some applications, they also presented computational challenges when applied large-scale graphs [4], [18], [49]. Later advancements shifted toward spatial methods that scale linearly in the number of nodes [117], directly defining convolutions through local neighbor interactions. These spatial techniques reduced computational complexity and made it easier to adopt strategies such as mini-batching and distributed learning, paving the way for the broad and efficient application of GNNs today.

A useful way to think about GNNs is by drawing comparisons to other deep learning architectures, such as CNNs and RNNs. Much like CNNs, GNNs incorporate local,"filter-like" mechanisms that capture neighborhood-level patterns. However, rather than working with a rigid grid as in images, the core operation in GNNs is message passing, where nodes collect and merge information from their neighbors to update their own embeddings. There is also a conceptual link to RNNs, in the sense that GNNs apply iterative updates that potentially resemble recurrence. Nevertheless, GNNs generally process layer by layer in a feed-forward fashion, rather than unrolling over time. Yet, issues like vanishing or exploding gradients can still appear in deeper GNNs, prompting researchers to introduce architectural enhancements, attention and gating modules aimed at maintaining stable training dynamics.

Over the years, numerous GNN variants have emerged, each creating unique methods for neighbor aggregation. GraphSAGE, for instance, learns a flexible aggregator which might perform averaging, max pooling, or even use an LSTM to handle sampled subsets of neighbors [144]. Meanwhile, the Graph Attention Network (GAT) employs attention mechanisms that build upon logical framework of letting each node assign different weights to its neighbors based on their importance [243]. Furthermore, it

allowed to mitigate and reduce computationally extensive matrix operations utilizing the graph structure itself. Graph Isomorphism Network (GIN) offers another perspective, shaping its aggregation rules to align with the expressive capabilities of the Weisfeiler-Lehman isomorphism test, thereby excelling at distinguishing nuanced structural differences in subgraphs [115].

Choosing how to train a GNN strongly depends on the intended goal. For instance, if node classification is the objective, one typically applies a cross-entropy loss [159] to a labeled subset of nodes, especially useful when many nodes in the graph remain unlabeled. In contrast, link prediction tasks revolve around assessing the probability that two nodes are connected by an edge. For problems requiring a classification decision for the entire graph, a readout or global pooling layer is used to aggregate all node embeddings into a single vector, which is then passed through a standard classification module. By customizing these procedures according to the nature of the underlying data, GNNs can learn rich, context-sensitive representations that excel in a variety of graph-focused applications.

Much like pooling strategies in CNNs that gather global context and reduce computational costs, specialized graph pooling algorithms have emerged to downsample nodes and edges. Approaches including Top-k pooling [27], [119], [143], SAGPool [119], [131], DiffPool [267], ClusterPooling [216], ASAPolling [199], PANPooling [158], and MemPooling [2] identify the most crucial subset of nodes in a hierarchical graph structure. This process parallels the way CNNs distill extensive images into progressively smaller feature maps, helping to simplify complex graphs into manageable layers of representation.

Beyond strictly supervised or semi-supervised applications, investigations into unsupervised and self-supervised methods for GNNs training are gaining traction. Inspired by the concept of autoencoders in CNN or RNN architectures, graph autoencoders implement an encoder-decoder pipeline to map a graph's adjacency or feature matrix to a latent space, then reconstruct it [118], [207]. Meanwhile, contrastive learning frameworks push node or subgraph embeddings to be similar when they hold related contextual information, offering a promising avenue for exploiting unlabeled data effectively [248], [268].

One advantage behind GNN development is that many real-world problems naturally form graphs [35]. Social networks, for instance, are a prime target: each person corresponds to a node, and edges indicate friendships or interactions [166]. Knowledge graphs, which store factual information via entities and their relationships, also benefit from GNN approaches to perform link prediction or entity classification [105]. Biological and chemical tasks (where proteins or molecules can be represented as nodes and bonds or interactions as edges) have similarly seen substantial improvements by using GNN models that can learn meaningful molecular substructures [202].

Graph Convolutional Networks have also shown great promise in Earth observation (EO) tasks, where spatial relationships and geospatial dependencies play a critical role. In EO, satellite imagery or remote sensing data can be modeled as

graphs, with nodes representing geographic regions or pixels, and edges encoding spatial adjacency or contextual similarity. Such approach was utilized in the PhD Candidate's work in [234]. GCNs have been effectively applied in land use and land cover classification [111], [285], environmental monitoring [102], crop type recognition and crop semantic segmentation [234]. By leveraging graph structures, these models can exploit both the spectral features from satellite images and the spatial correlations between neighboring regions—an advantage over traditional CNN-based approaches, which often fail to capture non-Euclidean spatial dependencies. Furthermore, GCNs facilitate semi-supervised learning [65], which is particularly beneficial in EO applications where labeled data are scarce or expensive to obtain.

Nevertheless, GNNs come with challenges as well. High-degree nodes can lead to a computational explosion, as message-passing might involve aggregating from thousands of neighbors. Many approaches address this by limiting the number of neighbors considered or by introducing hierarchical sampling strategies. Furthermore, graph size variability can complicate model design, since real-world graphs can range from small subgraphs of tens of nodes to massive networks with millions of nodes. Scaling GNNs up to such large graphs involves specialized data partitioning, mini-batching, and distributed training solutions [160].

Another critical consideration is the phenomenon of over-smoothing in very deep GNNs. As one can stack many message-passing layers, node embeddings can become progressively more similar, effectively losing the granularity needed for classification or other tasks. To counter this, researchers have developed skip connections, residual layers [147], or normalization strategies that preserve local distinctiveness [25], [98], [99], [132]. Gating functions similar to those seen in RNNs—like gating residual connections—can also be employed to carefully blend new information with existing hidden states, helping maintain meaningful variations in node embedding distributions [107].

Although GNNs are a relatively recent addition to the DL field, they have already attracted significant attention, partly due to their impressive success across numerous fields. From recommending new content in social or e-commerce platforms (using user-item interaction graphs) to predicting novel compounds in computational chemistry, GNNs have reliably showcased their capability [35], [105], [166], [202]. Dynamic or temporal GNN architectures push these boundaries even further by handling evolving graphs where nodes or edges change over time.

In a similar fashion to how sophisticated gating in RNNs can prolong the effective memory of sequences, GNNs can also be enhanced with memory modules for certain use cases, storing historical states of individual nodes or entire subgraphs [157]. This can be particularly advantageous in scenarios where the underlying relationships shift, as the network can adapt by blending old and new representations stored in memory. Research on interpretability and explainability for GNNs is also on the rise, with methods aimed at identifying the crucial subgraph structures or edges that lead to a given prediction [61], [97], [138].

Given their unique blend of representational power and flexibility, GNNs represent one of the most rapidly evolving niches in DL. They continue to inspire not only new architectures (e.g., those featuring advanced attention mechanisms [243] or spectral-based transformations [144]) but also fresh aforementioned application domains that were once thought too unstructured for neural network solutions. As the field matures, standard practices and deeper theoretical insights are helping to establish best-in-class approaches for training large-scale, high-performing graph models, setting the stage for further breakthroughs in graph representation learning, knowledge discovery, and beyond.

To shed more light on the mathematical foundation and operational awareness, we provide description of a GCNN layer. Each $(l+1)$-*th* layer in GCNN can be formulated as [117]:

$$H^{l+1} = \mathrm{ReLU}(\hat{A}H^{l}W^{l}), \tag{3.7}$$

where H^l and W^l denote the activation and trainable parameter matrices of layer index l. Within the initial layer of GCNN, i.e., $l = 1$, H^l presents the input neighborhood in matrix-based format, where the number of rows is equal to the number of samples, and columns define the number of features. Furthermore, $\hat{A}$ is the normalized adjacency matrix:

$$\hat{A} = D^{-\frac{1}{2}}AD^{-\frac{1}{2}}, \tag{3.8}$$

where D is the diagonal *degree matrix* obtained by summation of all columns of A per consecutive row i, and fills the value in $d_{i,i}$. We encompass the self-connections within the graph, thus each node can utilize its own features during the aggregation step. Such operation is obtained by summing the identity matrix to the adjacency matrix $\widetilde{A} = A + I$, and recomputing the degree matrix from $\widetilde{A}$. Finally, we obtain:

$$\hat{A} = \widetilde{D}^{-\frac{1}{2}}\widetilde{A}\widetilde{D}^{-\frac{1}{2}}. \tag{3.9}$$

In the Figure 3.6, we visualize a 2-layer GCNN without node reduction, i.e., 5 nodes in both layers. The first layer's nodes are depicted in blue, whereas the second layer is represented by red color. Within the first layer, the dimensionality of nodes is set as 16. Moreover, the second layer constitutes 32 features. To demonstrate feature aggregation, we include connections (highlighted in orange) that indicate the local neighborhood of 2nd node. Finally, we include dimensionality of weight matrix that is utilized in projection from 16 to 32 features.

To moderate the scale of the activations each node accumulates, it is a often practice to apply a normalization step on the adjacency matrix. This practice is especially helpful when certain nodes exhibit a large number of connections, as it averts excessively large representations for those "center" nodes and comparatively diminished ones for "border" nodes—imbalances that can trigger vanishing or exploding gradients [85]. By normalizing the adjacency matrix, each node's embedding becomes a weighted average of itself and its neighbors, thus contributing to more stable training.

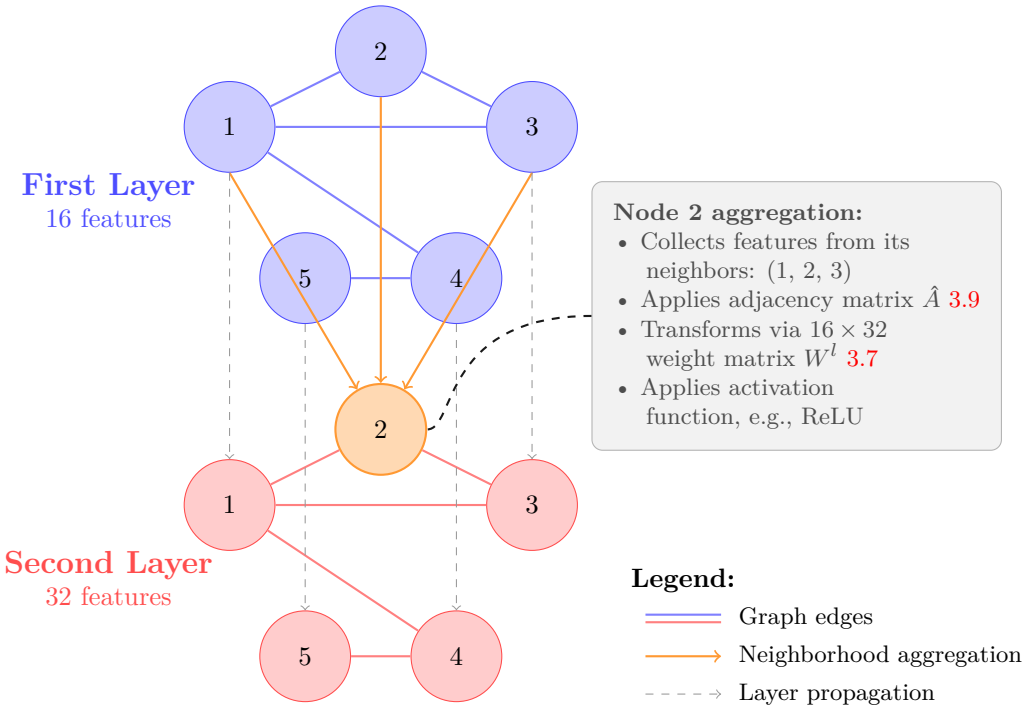


FIGURE 3.6: Graph Convolutional Neural Network showing neighborhood aggregation mechanism. The highlighted node in the output layer demonstrates how GCN layers aggregate information from neighboring nodes in the input layer, following the Kipf & Welling (2017) formulation.

Moreover, in the $(l+1)$-th layer of the network in GCNN, an $(l+1)$-hop connectivity concept is preserved [127]. In the second layer, the aggregation process grabs features not only from a node's immediate neighbors but also from the neighbors of those neighbors. This expanded scope enriches the contextual awareness of GCNN, enabling it to identify more abstract patterns in both spatial and temporal dimensions.

Chapter 4

Related Literature

Due to the unprecedented success of ML and DL, there is a wide range of SOTA methods that leverage the advantages of ML to automatically extract features and predict the desired output target in a wide range of various applications [128]. Traditional approaches often rely on standard techniques that use domain-specific or hand-crafted features. Conversely, in DL, the user is not required to possess *a priori* knowledge, which can be particularly challenging in highly constrained environments. Instead, we benefit from automated representation learning which may indeed help us capture features that were not possible to define by hand.

Naturally, when depending on ML systems, it is crucial to consider the significance of explainability, which often diminishes as model complexity increases. DL models, in particular, are frequently characterized as *black-box* systems due to their high dimensionality and non-linear internal representations, making their internal decision processes difficult to interpret directly from a human or scientific standpoint. However, the field of explainable artificial intelligence (XAI) [56] has advanced significantly in recent years, offering a variety of tools and techniques—such as SHAP [156], LIME [204], Integrated Gradients [222], or Grad-CAM [211]—that aim to uncover and visualize how these complex models arrive at their predictions. While full transparency remains a challenge, these methods provide valuable insights into model behavior, allowing researchers and practitioners to better understand, debug, and trust ML systems.

Since this thesis addresses the problem of HU, this chapter will focus on DL approaches applied to abundance estimation in the context of remote sensing. Given the strong conceptual and methodological overlap between HU and HSI segmentation (HS), the literature review will encompass both domains. Although the primary emphasis is placed on modern ML and DL-based methods, traditional and baseline techniques will also be discussed to provide broader context and illustrate the field's evolution. Importantly, both HU and segmentation tasks typically operate on homogeneous hyperspectral data, allowing for architectural designs developed for segmentation to be effectively adapted for unmixing purposes.

Hyperspectral segmentation and unmixing, like many other remote sensing tasks,

can be approached using supervised, unsupervised, or reinforcement learning methods. In supervised learning, datasets include labels provided *a priori*, allowing models to learn mappings between inputs and corresponding outputs. After training, the model is evaluated on a separate, previously unseen test set to assess its generalization performance.

By contrast, unsupervised learning operates without explicit labels or ground-truth annotations. Instead, models aim to discover the intrinsic structure of the data, such as underlying clusters or patterns in the feature space. This category includes techniques such as clustering, dimensionality reduction, density estimation, and anomaly detection. Since no labeled supervision is used, there is generally no strict division between training and testing sets.

A third paradigm, reinforcement learning (RL), has also begun to gain attention in remote sensing contexts. In RL, an agent learns to take sequential actions in an environment to maximize cumulative rewards. While RL has traditionally been applied in control and robotics, its emerging applications in hyperspectral analysis include adaptive sampling, sequential band selection, and iterative label refinement. These methods are particularly useful in scenarios where decisions are made over time and involve trade-offs, such as balancing data fidelity with computational cost.

4.1 Machine Learning for Hyperspectral Segmentation

HS, being pixel-level classification, has been thoroughly researched in the literature due to its practical applications in Earth observation. HSI segmentation via Kernel Sparse Representation (KSR) proposed in [37] is a method designed to improve pixel classification accuracy by leveraging the power of sparse coding combined with kernel methods. Instead of directly performing unmixing, KSR focuses on representing each hyperspectral pixel as a sparse linear combination of training samples mapped into a high-dimensional feature space via a kernel function. This approach captures nonlinear relationships within the data, which are common in hyperspectral imagery, enabling better discrimination between classes. The sparse coefficients obtained serve as features for classification, typically through a classifier such as support vector machines or nearest neighbors. By integrating sparsity and kernel-based nonlinear mapping, KSR effectively enhances segmentation and classification performance in hyperspectral images, particularly in complex scenarios where spectral signatures exhibit nonlinear mixing or variability.

In [257] the authors introduce a ML-based method for automated segmentation of scanning electron microscope (SEM) images of organic-rich shale samples. Their approach combines classical feature extraction techniques—such as wavelet transform, Gaussian blur, and Hessian matrix features—with a random forest classifier to accurately identify and segment four key shale components: pores/cracks, matrix, pyrite, and organic/kerogen. The method addresses challenges posed by overlapping pixel intensities that hinder traditional threshold-based segmentation, leveraging spatial and

multiscale information for improved differentiation. The authors demonstrate that their approach achieves high segmentation accuracy and operates efficiently on large SEM images. This work highlights the robustness of combining feature engineering with supervised learning for material characterization in geoscience applications.

The authors in [122] present a novel semi-supervised ML method for pixel classification and segmentation of mid-infrared HSI tissue samples. Their approach leverages both labeled and unlabeled pixels for spectral dimension reduction and hierarchical clustering, addressing the challenges posed by noisy and high-dimensional HSI data. Unlike traditional supervised methods that require extensive annotations, their technique effectively handles limited labeled data while capturing the variability within sub-cellular components. The method achieves a strong classification performance with an F1 score of over 71% on cross-validation across multiple tissue images. The authors also release their code to promote further research and demonstrate that disease classification can be simplified following accurate HSI segmentation.

In [135] the authors propose a semisupervised segmentation algorithm tailored for high-dimensional HSI data. Their approach consists of two main stages: first, semisupervised learning of the posterior class distributions using multinomial logistic regression, where the model parameters are estimated from both labeled and actively selected unlabeled samples via a graph-based method. Second, the segmentation step integrates spatial contextual information by applying a multilevel logistic Markov random field prior to enforce spatial smoothness in the labeling. The final segmentation is computed through an efficient maximum a posteriori estimation. Experimental results on both synthetic and real hyperspectral datasets demonstrate that their method achieves classification accuracy comparable to or better than existing supervised techniques, highlighting the advantage of combining spectral and spatial information within a semisupervised framework.

4.2 Deep Learning for Hyperspectral Segmentation

In [208], the authors proposed using spectral-spatial convolutional operations to enhance segmentation performance. The architecture of their model is organized into three main components. The initial block employs a 1×1 spatial convolution, targeting the extraction of spectral features from the local image region. This block's primary role is to extract features from the input patch and modify the number of bands to a value divisible by the number of blocks defined in the subsequent stage. This adjustment is accomplished by splitting the activation map into n consecutive non-overlapping band sets. The second block comprises parallel spectral convolutional layers that operate across the spatial axis. Its objective is to capture both spectral and spatial information while keeping the model's complexity low by enabling parameter sharing across the parallel modules. Finally, the third block integrates multiple fully connected layers, which process the concatenated activation maps output by the parallel modules in the second block.

Empirical evaluations demonstrate that this model achieves precise segmentation and surpasses other approaches such as k-Nearest Neighbor (k-NN) [283], Support Vector Machine (SVM) [45], Multilayer Perceptron (MLP) [187], and an alternative CNN model described in [95]. Furthermore, it introduces an innovative method that combines the pixel-pair technique to augment the training dataset, employing CNN for feature extraction and a voting mechanism for inference [137].

In the study presented in [95], the authors proposed a simple architecture that integrates a convolutional layer followed by a max pooling operation, which acts as a feature extractor, and concludes with two fully connected layers that function as the classifier. In contrast to previous methods, this approach exclusively processes input samples that contain only the spectral dimension of a specific pixel in HSI, omitting any spatial information. The experimental results demonstrate that this streamlined architecture can surpass the performance of other models, including LeNet-5 [130], which encompasses two convolutional layers. This is particularly evident in situations where training sample availability is severely limited. Such findings are especially relevant in remote sensing applications, where restricted training data is frequently encountered, notably in HU. These results underscore the advantages of employing simpler architectures in DL approaches for these challenging scenarios.

An approach presented in [165] employs RNNs for feature extraction along the spectral dimension. This research also introduces a new activation function known as the parametric rectified hyperbolic tangent, which the authors claim facilitates higher learning rate parameters. This modification accelerates the training process and reduces the likelihood of divergence or overshooting the minimum. The authors assert that this study represents the first use of RNNs for HSI segmentation. Experimental results indicate that this model, in conjunction with CNNs, achieves high-quality land cover classification.

The study described in [3] introduced an approach to hyperspectral segmentation that combines superpixels with CNNs. The method involves segmenting the HSI into superpixels, which represent distinct regions, followed by labeling these regions using a deep CNN. This process integrates both spectral and spatial information. Experimental findings indicate that applying segmentation prior to labeling can significantly improve the overall segmentation accuracy.

In [70], the authors implemented CNNs combined with multiple feature learning techniques to enhance model accuracy. The architecture they proposed is composed of two main parts: a feature extractor and a predictor. The feature extractor includes several parallel convolutional blocks, allowing for the extraction of varied feature representations by merging spectral and spatial information from the input sample. These features are then concatenated and forwarded to the predictor, which utilizes a softmax activation function to compute the probability distribution of class memberships for each pixel. The experimental results indicated that the integration of multiple features significantly boosts the performance of the models.

The study presented in [255] introduced a convolutional-RNN specifically designed

for hyperspectral pixel segmentation. This architecture effectively merges the benefits of convolutional and recurrent operations, resulting in enhanced segmentation performance when compared to other advanced DL models. The network is organized into two primary components: the first part consists of several convolutional layers, which are followed by recurrent layers. Ultimately, the prediction of labels is accomplished using the softmax function, which is based on the features extracted from the final recurrent layer.

The work presented in [151] proposed a DL-driven method for band selection, utilizing CNNs integrated with an attention mechanism. The approach also incorporates anomaly detection to pinpoint the most informative spectral bands. Experimental findings revealed that this method maintains segmentation accuracy while effectively identifying the key regions of the spectrum. On top of that, the underlying CNN model equipped with attention modules, can be utilized after training for precise HSI segmentation.

In [173], the authors evaluated the generalization capabilities of CNNs by simulating various atmospheric conditions and noise distortions. Their experimental findings revealed that certain undesirable image artifacts can significantly impair segmentation quality, underscoring the necessity of simulating such noise to create models that are well-generalized for real-world applications. Furthermore, they verified the robustness of such algorithms against data-level distortions that correspond to real-life phenomenon which is critical for on-board applications for EO tasks.

In [181], the authors introduced an approach for HSI segmentation that combines multiple 3D convolutional layers with various augmentation techniques to tackle the issue of limited ground truth (GT) data. The experimental results demonstrated that this method surpasses the performance of other SOTA techniques.

TABLE 4.1: Overview of hyperspectral segmentation algorithms.

Year	Ref.	Method	Type	Datasets	Metrics	Approach
2017	[208]	Band-adaptive spectral-spatial feature learning neural network for HSI segmentation.	DL	Indian Pines, Pavia University, Salinas	Class-Specific Accuracy, Overall Accuracy, F-Score, Kappa Coefficient	Spectral-Spatial
2015	[95]	1D-CNN with max pooling and fully-connected layers	DL	Indian Pines, PAvia University, Salinas	Overall Accuracy	Spectral
2017	[165]	RNN with gating mechanisms and parametric rectified hyperbolic tangent activation function.	DL	Indian Pines, Pavia University, Houston	Overall Accuracy, Average Accuracy, Kappa Coefficient	Spectral
2016	[3]	Superpixels with CNN are utilized to perform HSI segmentation.	DL	Indian Pines	Overall Accuracy, Average Accuracy	Spectral-Spatial
2018	[70]	CNN with multiple feature learning and parallel blocks to enhance HSI segmentation.	DL	Indian Pines, Pavia University, Salinas	Overall Accuracy, Average Accuracy, Kappa Coefficient	Spectral-Spatial

2017	[255]	Convolutional RNN to learn more discriminative features for HSI segmentation.	DL	Indian Pines, Houston	Overall Accuracy, Average Accuracy	Spectral-Spatial
2020	[151]	CNN with attention modules for band selection and segmentation.	DL	Indian Pines, Pavia University, Salinas	Overall Accuracy, Average Accuracy, Kappa Coefficient	Spectral
2021	[173]	Atmospheric noise simulation and generalization capabilities of spectral and spectral-spatial CNNs.	DL	Indian Pines, Pavia University, Salinas, Houston	Overall Accuracy, Average Accuracy, Kappa Coefficient	Spectral and Spectral-Spatial
2020	[181]	3D-CNN with data augmentation to enhance quality of HSI segmentation.	DL	Indian Pines, Pavia University, Salinas	Overall Accuracy, Average Accuracy, Kappa Coefficient	Spectral-Spatial

4.3 Machine Learning for Hyperspectral Unmixing

One of the earliest and most prominent methods for HU endmember extraction is the *N-FINDR* algorithm [254]. This method operates by identifying the set of pixels that form a simplex with the maximum volume within the data space, under the assumption that the true endmembers correspond to the vertices of this simplex. By maximizing the volume of the simplex, N-FINDR effectively finds the most spectrally pure pixels that define the data distribution, making it well-suited for hyperspectral images where pure material signatures are assumed to be present.

Another widely used algorithm for endmember extraction is the *Pixel Purity Index* (PPI) [21]. PPI works by projecting the hyperspectral pixels onto multiple randomly generated vectors and counting how often each pixel appears at the extremes of these projections. Pixels that frequently appear as extremes are considered spectrally pure and likely to be endmembers. This approach is computationally efficient and robust to noise, and has been extensively applied in remote sensing for material identification.

In [51], authors propose a novel blind hyperspectral unmixing algorithm based on an extended linear mixing model (LMM) to effectively address spectral variability. Their approach introduces a pixelwise, spatially coherent local variation of endmembers by allowing scaled versions of reference spectra, overcoming limitations of the classic LMM which assumes fixed endmember signatures. The method unifies several existing techniques under a common framework and demonstrates superior performance on both synthetic and real datasets, accurately estimating endmember variability and fractional abundances in complex hyperspectral scenes.

The authors in [247] introduced a hypergraph-regularized sparse nonnegative matrix factorization (NMF) algorithm for hyperspectral unmixing. By modeling the spatial–spectral joint structure through a hypergraph, their approach captures high-order similarity relationships among spatially neighboring pixels, which enhances the estimation of abundances. This method improves upon existing sparsity-constrained NMF models by enforcing abundance consistency within hyperedges, leading to more robust unmixing results. Experimental evaluations on synthetic and real hyperspectral datasets demonstrate the superiority of the proposed algorithm compared to several state-of-the-art techniques.

4.4 Deep Learning for Hyperspectral Unmixing

In [279], the authors introduce a CNN architecture that combines multiple convolutional layers followed by several fully connected layers. The initial layers function as a feature extractor, while the final layers use the extracted feature vectors to estimate the abundance of each endmember in the HSI. To mitigate the risk of overfitting, the model employs the dropout technique [218] that randomly zeros activations within the model's layers. Experimental results demonstrated that this approach outperforms other methods, such as linear spectral unmixing [88] and a cascade DNN-based

model presented in [141]. Given its innovative nature and high-quality unmixing performance, this model is among those investigated in this thesis and will be explored in greater detail in chapter summarizing our experimental studies.

In [141], the authors proposed a two-step methodology based on artificial neural networks. In the initial phase, a network is trained to represent the input pixel in a reduced dimensional space. This model can be divided into two parts: an encoder and a decoder, which function sequentially. This design is often referred to as an autoencoder, where the encoder learns to map the input data to a compact representation, referred to as the code. The decoder then takes this feature vector and reconstructs it into the original space. In the second phase of training, the pre-trained encoder is used along with an additional multilayer perceptron (MLP) in a sequential manner. Once the input passes through the encoder, the MLP predicts the abundances of the endmembers in the hyperspectral image. The experimental findings demonstrated that this approach delivers superior unmixing performance compared to linear spectral unmixing.

Another interesting approach in HU was presented in [113]. The authors proposed a DCAE model. The functionality of the model is analogous to the former ANN model. The encoder part contains convolutional layers, whilst the decoder's weights are set to the endmember matrix. Such operation allows the authors to linearly rebuild the endmember fraction vector by matrix multiplication with the endmember matrix. The endmember matrix contains information about the values for each hyperspectral band for each considered material. To alleviate overfitting, the model contains dropout [218]. To fit to the unmixing scenario, the objective function utilized is spectral information divergence.

In [239], we explored how the size of the training set affects the performance of HU using two models previously discussed in [113], [279], namely the CNN and DCAE architectures. The experiments, conducted on two real-life datasets, showed that the unmixing accuracy drops noticeably when the training set is small. Moreover, a certain point was found where increasing the number of training samples no longer leads to significant improvements in the unmixing results. These findings suggest that identifying this threshold could help reduce the costs involved in manual labeling.

In [219], the authors propose a novel architecture that builds upon a deep autoencoder framework. This architecture is structured into two key components. The first component, which consists of stacked autoencoders, is designed to extract features efficiently while maintaining robustness against outliers. Meanwhile, the second component utilizes a variational autoencoder to estimate both the endmember signatures and their corresponding abundance fractions. Experimental assessments conducted on synthetic and real datasets indicate that this approach achieves performance levels comparable to other leading methods in HU.

In [188], the authors introduced an autoencoder-based method, where the encoder is made up of four fully connected layers, and the decoder's weights are set to the endmember spectra. The code, which refers to the activations from the encoder's

final hidden layer, serves as the abundance vector for the input sample. To guarantee nonlinear mapping within the encoder, several nonlinear activation functions are employed, including Sigmoid [186], ReLU [170], and LReLU [260]. Additionally, to accelerate the training phase, batch normalization [99] is applied to one of the encoder layers. Lastly, to prevent overfitting and introduce regularization, a Gaussian Dropout layer [218] is added at the conclusion of the encoder module.

In [189], the authors introduced a novel approach to hyperspectral unmixing using a multitask learning model built on parallel autoencoders. This architecture incorporates both spectral and spatial features to exploit the relationships between neighboring pixels. Each pixel is processed through its own encoder path, and the decoder applies a linear mixing model. The results from each path are then combined and passed through a shared hidden layer, which helps capture the contextual information of adjacent pixels. The method was tested on two real-world datasets, and the results demonstrated its superior performance compared to other methods evaluated in the study.

The authors of [60] propose a novel autoencoder (AE)-based neural network for unsupervised hyperspectral unmixing, leveraging the multilinear mixing model (MLM). Unlike traditional linear or bilinear models, MLM models light reflections through discrete Markov chains, capturing infinite degrees of endmember interactions. The network architecture explicitly incorporates endmembers, abundances, and transition probabilities, functioning in two configurations: MLM-1DAE, which analyzes pixel-wise spectral data, and MLM-3DAE, which incorporates spectral-spatial correlations. Furthermore, the decoder segment is composed of three parts, each one utilizes linear or non-linear characteristics coupled with reconstruction of the input spectra. Experimental results on synthetic and real datasets show the model's competitive performance compared to traditional MLM-based methods.

In [74], the authors examine the utilization of autoencoder models for the purpose of unmixing in the context of non-destructive molecular composition. The experiments proof that such methodology provides superior accuracy and robustness compared to conventional techniques, e.g., involving traditional chemometric methods. Moreover, such approach could be extrapolated to unmixing in the case of EO data.

Furthermore, in [124], a novel algorithm for HU was introduced. It incorporates multivariate curve resolution with alternating least squares and state-of-the-art autoencoder models. The proposed approach allowed to reduce the cost of manual work and accurate detection of solid waste. In this study, solid waste refers to heterogeneous mixtures of discarded materials—including leather, paper, wood, plastics, textiles, inert waste, and food waste—often contaminated with liquids such as water, oil, and leachates, whose complex composition challenges spectral analysis and identification.

In [224], the authors provide a new algorithm for HU which encapsulates a dual-feature fusion network that is composed of four parts. It incorporates feature fusion

module, abundance estimation, endmember estimation, and lastly, the reconstruction module. The approach was evaluated on synthetic and real data, proving high precision in the context of unmixing. Furthermore, the model also can extract endmember signatures and reconstruct images, thus provides a cross-task functionality.

Another method which incorporates an autoencoder based approach was proposed in [220]. The method is based on multilinear mixture model, which takes into account multiple scattering problem. Such phenomenon occurs when electromagnetic waves encounter multiple particles on their way to the observer object. The model incorporates autoencoder-based augmented network. The encoder part is composed of two modules, where the first one focuses on the linear spectrum, whilst the latter targets the problem of multiple scattering. The decoder reconstructs the input sample.

In [264] a new method for HU was introduced that incorporates NMF as the foundation algorithm. The authors proposed to utilize NMF to enhance the accuracy of HU. To improve robustness of this process, NFM was adapted to a low-rank and sparse variation. The experimental results provided that such method allows for high quality endmember and abundance estimation.

In [262], a DL-based architecture was introduced to tackle HU. The model incorporates convolutional layers with stationary wavelet transform to improve noise robustness in the spatial-spectral feature domain. The proposed innovation of standard convolutional layers with injection of advanced feature learning strategy allowed to surpass other baseline methods over simulated as well as real benchmark HU datasets.

The authors of [246] propose a new DL model called MAT-Net for HU, addressing the limitations of existing methods that extract spatial-spectral features. To improve detection of heterogeneous ground objects, and avoid incomplete or mixed features, MAT-Net uses an encoder-decoder architecture that combines spectral and spatial data. It includes a dual-stream, multibranch CNN encoder to capture both spectral and multiscale spatial features, with a block-by-block branching strategy to optimize computational cost. Additionally, a transformer encoder with multihead self-scale-aggregation attention blocks is introduced to improve feature integration by adapting to different scales. Experimental results show that MAT-Net outperforms existing methods in both synthetic and real datasets.

The authors of [225] proposed a novel method to tackle HU that incorporates attention mechanism to enhance the process of abundance estimation utilizing convolutional layers in the spectral as well as spatial domain. Furthermore, the authors utilize various kernel shapes (such as 1×1 and 3×3), to enhance the process of feature extraction. From the experimental results over benchmark datasets, the approach proved to execute on a par with state of the art approaches.

Another approach utilized an evolutionary algorithm (EA), to unmix the hyperspectral pixels [134]. The authors proposed a multi-step approach which utilizes a weakly non-dominated sorting algorithm to tackle various problems encountered in HSI analysis. The algorithm decomposes an input image into homogenous regions,

which later undergo multiobjective optimization. Such operation allows to alleviate the problem of dimensionality which often occurs in hyperspectral imagery. The experiments proved that such method allows to better tackle the curse of dimensionality [6], as well as allows for better convergence in terms of the final solution quality.

The authors of [31] proposed a model with a deep shared fully connected autoencoder, which incorporates feature fusion with two different branches in the input, i.e., the spectral dimension and spectral features from superpixel segmentation. The latter approach instead of utilizing square-shaped neighborhoods, incorporates a hexagon structure for each superpixel. The experimental results provided that such non-standard grid structure allows to obtain accurate abundance estimations over several real datasets.

In [72], the authors proposed a reversible generative network for the problem of HU. The model incorporates an endmember learning module to understand the process of endmember generation as well as an abundance guidance module to estimate the abundances of targets in the imagery. The experimental results provided that such architecture allows to understand the endmembers as well as abundances and tackle the challenge of spectral variability.

Another interesting approach which utilizes an autoencoder architecture and incorporates transformer-like methodology was proposed in [73]. The model exploits spatial and spectral features which undergo fusion by incorporating local convolution operations as well as a transformer mechanism to understand global-wise relationships. The architecture includes a spectral Swin Transformer [148] and spatial convolution blocks. the proposed model proved to be robust against noise and provide state-of-the-art accuracy and outperformed other HU methods.

In [16] a transformer-based architecture was proposed to tackle HU. The approach incorporates Principal Component Analysis (PCA) and spectral-spatial preprocessing step to extract both features from the input sample. The underlying architecture uses a vision transformer to improve the abundance extraction capabilities. The model proved to achieve high quality results over three real benchmark datasets.

Another research introduced a multiscale convolutional mask network for HU [263]. The model utilizes two main modules, i.e., the mixed region mask and a multiscale convolution AE. Furthermore a new strategy for initialization is used which incorporates vertex component analysis [183] and density-based spatial clustering of application with noise [59] to alleviate the problem of outliers. The introduced method allowed the authors to improve the SOTA methods and provide accurate endmember and abundances estimations over real and simulated unmixing benchmark datasets.

In another research [53] a novel DL-based approach was introduced. It incorporates a double-aware transformer model for HU. It extracts features from both dimensions, i.e., spatial and spectral axes. The architecture includes score-based homogenous-aware module and spectral group-aware module to execute the aforementioned feature fusion and enhance the estimated abundance maps. Experiments

over real and simulated benchmark datasets proved that the model delivers high-quality unmixing.

Authors in [265] proposed a novel architecture, i.e., an U-shaped transformer model with shifted windows. It incorporates spatial features with the use of multihead attention blocks and more importantly, operates over the entire image, without the use of patching during preprocessing of input samples. Furthermore, the method utilizes downsampling and upsampling with various scale factors, to introduce more informative representation for feature fusion. The experimental results provided that the model surpasses other DL architectures and allows to obtain accurate abundance maps.

In [198] the authors proposed a method to enhance graph learning capabilities for HSI analysis in the context of HU by utilizing higher-order graph regularizer with nonnegative matrix factorization coupled with adaptive feature selection. In the graph structure, the second-order nearest-neighbor relations with adaptive weighting were incorporated. The proposed method was validated on real and simulated datasets demonstrating its high precision and superiority over baseline solutions.

The authors in [197] proposed another method based on nonnegative matrix factorization with fast sparse modification to overcome the issue of nonconvex and nonsmoth optimization. The algorithm uses gradient-based modifications to solve local subproblems within the nonnegative matrix factorization method. Based on the experimental results, the approach demonstrated superiority over other algorithms.

The authors of [133] proposed a DL architecture to tackle the task of HU by exploiting a semi-supervised framework. The model utilizes a cube-based input with a 3D convolutional AE network. It extracts spatial and spectral features through the use of the attention mechanism to improve the abundance estimation capabilities. The abundance maps estimated inside the model are treated as latent features in the AE structure, and are fed to the regression module. The proposed architecture proved to outperform other convolutional-based methods.

In [94], a interesting HU model was introduced. It can be decomposed into two main parts, an endmember unmixing network, related to endmember extraction and processing, and the abundance unmixing network, used for abundances extraction. The first module incorporates a variational autoencoder (VAE) coupled with a multi-scale convolution block attention module with pretraining. The latter abundance module uses a multi-scale attention convolution block to incorporate spectral and spatial attention features to enhance the unmixing capabilities. The experimental results showed that such architecture allows to improve and outperform the SOTA models.

In [106], the authors proposed a new graph attention convolutional AE model to tackle the task of HU. The model focuses on long-range and short-range spatial dependencies by employing graph attention convolutions. Such operation allowed the authors to enhance the unmixing capabilities of the proposed AE model. The decoder part is constructed as a nonlinear mixing model to allow for more complex physical

interpretability and reconstruction. The experimental results over real and simulated data indicated that such model is able to outperform the SOTA methods in the task of HU.

The authors in [50] proposed an approach utilizing block-term decomposition models for HU. The model aims to overcome the current limitations of decomposition methods. It incorporates a two-factor reparametrization of tensor (it breaks down a complex multi-dimensional tensor into two simpler factors, these two factors correspond directly to the endmember components and the abundance components), which aligns with the endmember and abundance components. This method allows to inject physics-based priors and adjust to specific HU conditions. The optimization problem is solved by using the gradient projection algorithm. The experimental results provided that such method allows for fast convergence over the compared baseline solutions.

In [217] the authors proposed a first-order graph trend filtering method for HU. The goal of the architecture is to enhance the spatial features by utilizing first-order graph difference operator from an original input hyperspectral cube. Furthermore, the authors increase the sparsity of abundances by incorporating L1 regularization [184]. The experimental results showed that such a method allows for precise unmixing and abundance estimation over real and simulated datasets and allows to overcome the existing baseline solutions.

TABLE 4.2: Overview of hyperspectral unmixing algorithms.

Year	Ref.	Method	Type	Datasets	Metrics	Approach
2011	[141]	Cascade ANN Model	DL	CHRIS-PROBA, AVIRIS, AHS	MSE, RMSE	Spectral
2018	[279]	CNN-based architecture	DL	Jasper Ridge, Urban	RMSE, rmsAAD	Spectral-Spatial
2018	[188]	Autoencoder-Based Method	DL	Jasper Ridge, Urban, Samson	MSE, SAD	Spectral
2019	[219]	Deep Autoencoder Framework	DL	Mangrove, Samson, Cuprite	RMSE, SAD	Spectral
2020	[113]	Deep Convolutional Autoencoder	DL	Jasper Ridge, Urban	RMSE	Spectral-Spatial
2020	[189]	Multitask Learning Model	DL	Urban, Samson	MSE, SAD	Spectral-Spatial
2021	[239]	Training Size Analysis	DL	Jasper Ridge, Urban	RMSE, rmsAAD	Spectral-Spatial
2023	[53]	Double-Aware Transformer Model	DL	Jasper Ridge, Samson, Apex	RMSE, SAD	Spectral-Spatial
2023	[265]	U-Shaped Transformer Network	DL	Jasper Ridge, Samson, Washington DC	RMSE, SAD	Spectral-Spatial
2023	[198]	Graph Regularized Nonnegative Matrix Factorization	Traditional	Jasper Ridge, Urban, Samson, Cuprite	RMSE, SAD	Spectral

2023	[197]	Fast Sparse Nonnegative Matrix Factorization	Traditional	Urban, Cuprite	RMSE, SAD	Spectral
2023	[133]	Semi-Supervised Attention-Based 3D convolutional Autoencoder	DL	Jasper Ridge, Urban, Shenyang	RMSE, AAD	Spectral-Spatial
2023	[94]	Multi-Scale Convolution Attention Network	DL	Jasper Ridge, Samson, Apex	RMSE, SAD	Spectral-Spatial
2023	[106]	Graph Attention Convolutional Autoencoder Model	DL	Jasper Ridge, Samson	RMSE, SAD	Spectral-Spatial
2023	[50]	Fast and Structured Block-Term Decomposition for HU	Traditional	Urban, Terrain	MSE, SAD	Spectral
2023	[217]	First-Order Graph Trend Filtering for Sparse HU	Traditional	Cuprite	RMSE	Spectral-Spatial
2024	[60]	Multilinear Mixing Model-Based Autoencoder	DL	Samson, Cuprite, Washington DC	RMSE, SAD	Spectral-Spatial
2024	[74]	Autoencoder for Molecular Composition	DL	240 Sugar Mixtures (In-House)	RMSE, SAD	Spectral
2024	[124]	Multivariate Curve Resolution—Autoencoder Algorithm	DL	NIR HSI of Waste Fractions (In-House)	SAD, RMSE	Spectral

2024	[224]	Dual-Feature Fusion Network	DL	Jasper Ridge, Urban, Samson	SAD, RMSE	Spectral-Spatial
2024	[220]	Autoencoder with Multilinear Model	DL	Moffett Field, Jasper Ridge, Shuangfeng, Samson	SAD, RMSE	Spectral
2024	[264]	Low-Rank and Sparse NMF Model	Traditional	Jasper Ridge, Urban, Samson	MSE, RMSE, SAD	Spectral
2024	[262]	Stationary Wavelet Convolutional Network with Generative Feature Learning	DL	Samson, Houston, Moffett	RMSE, SAD	Spectral-Spatial
2024	[246]	Multiscale Aggregation Transformer Network	DL	Jasper Ridge, Urban, Samson, Apex	RMSE, SAD	Spectral-Spatial
2024	[225]	Abundance-Guided Attention Network	DL	Jasper Ridge, Urban, Samson	RMSE, SAD	Spectral-Spatial
2024	[134]	Evolutionary Multiobjective Multitasking Optimization	Traditional	Cuprite	AAD	Spectral
2024	[31]	Deep Shared Fully Connected Autoencoder	DL	Jasper Ridge, Urban, Samson	RMSE, SAD	Spectral-Spatial
2024	[72]	Reversible Generative Network	DL	Samson, Cuprite	MSE, SAD	Spectral
2024	[73]	Transformer-Enhanced CNN Model	DL	Urban, Samson	SAD	Spectral-Spatial

2024	[16]	Convolutional Vision Transformer	DL	Jasper Ridge, Samson, Apex	RMSE, SAD	Spectral-Spatial
2024	[263]	Multiscale Convolutional Mask Network	DL	Urban, Samson, Washington DC	RMSE, SAD	Spectral-Spatial

4.5 Summary of Literature Review

Upon examining the existing literature on hyperspectral segmentation and unmixing, it becomes evident that these two challenges involve analogous type of data and are intrinsically connected. The primary distinction between the two lies in the type of outputs they generate. Consequently, models developed for one task can often be adapted to the other. For example, the feature extraction components of an existing model could be reused with minimal changes, such as altering the output layer or the activation function. By reviewing and summarizing the current SOTA methods in both fields, a collection of potential models that could be adapted for both tasks was identified in Table 4.1 (segmentation), and Table 4.2 (unmixing). Furthermore, transfer learning offers a alternative viable strategy, allowing a pre-trained model for segmentation to be fine-tuned for unmixing by modifying only the final output layer. Typically, in such transfer learning scenarios, the pre-trained weights of the base model remain fixed, and only the last layer is retrained, significantly reducing the time required to train the new model. Based on these points, it is apparent that models designed for segmentation and unmixing can be effectively adapted and applied across both tasks.

Chapter 5

Proposed Hyperspectral Unmixing Framework

5.1 Investigating the Impact of Varying Training Set Sizes on Hyperspectral Unmixing

In this section, we focus on the problem of the lack of labeled ground-truth data in the task of HU. As already mentioned, acquiring new, high-quality ground-truth datasets is extremely costly, user-dependent and infeasible in lots of practical scenarios. Thus, building machine learning models from limited ground truth is of paramount practical importance nowadays. It is one of the most critical problems in this domain since it hinders the possibility of developing and deploying DL-based methods, which require extensive amount of labeled training data. We investigate the impact of varying training set size on the quality of HU provided by DL-based models with various architectures, to quantify the impact of the size of such training sets on the capabilities of the corresponding models.

5.1.1 Deep Learning Models

We investigate two state-of-the-art models to estimate the abundance vectors of underlying endmembers. To ensure variability of models, we selected two topologically different architectures: *i*) a CNN model to employ its capability of learning hierarchical spectral-spatial patters and features, and *ii*) a Deep Convolutional Autoencoder (DCAE) which is based on an autoencoder concept. Such architecture uses the encoder and decoder that work in back-to-back fashion—the encoder learns how to properly map the input data into dense latent representation, whereas the decoder reconstructs the input sample given the latent compressed representation. The main objective of the study is to understand and measure the impact of varying training set size in the task of HU on the performance of different DL models.

The formerly mentioned CNN was proposed in [278]. The approach focuses on employing CNN architecture to utilize the automated feature- and pattern-learning capabilities of convolutional layers. The model is presented in two variations: *i*) the spectral approach which focuses on the analysis of a single pixel only (without its neighborhood) during the unmixing process, hence extracts only its spectral features,

and *ii*) spectral-spatial approach, to additionally enhance the unmixing quality by adding spatial features to the input context. The spatial features incorporate a local neighborhood of pixels, such a patch is centered on the element of interest for which the abundance vector has to be estimated.

The pixel-based variant employs a $1 \times 1 \times \lambda$ sample, where λ denotes the number of hyperspectral bands. Furthermore, it employs four convolutional layers where the kernels dimensionalities are set to $1 \times 1 \times 5$, $1 \times 1 \times 4$, $1 \times 1 \times 5$, $1 \times 1 \times 4$ for each layer, respectively. In each layer, the number of feature maps is set to 3, 6, 12, and 24. Moreover, the model is equipped with additional pooling layers to lower the dimensionality of the activation maps and control the gradient flow. Each convolutional layer utilizes a pooling kernel of shape $1 \times 1 \times 2$. Finally, the model is concluded with three fully-connected layers with 192, 150 and c units, respectively. Here c denotes the number of endmembers in the HSI. All of the layers employ ReLU non-linearity and the output is concluded with softmax function, to satisfy unmixing constraints. In the Figure 5.1, we depict the architecture of the pixel-based CNN.

The cube-based model enhances the feature-learning capabilities by applying a 3D sample with the following shape: $3 \times 3 \times \lambda$. Similarly to the pixel-based model, it employs four convolutional layers where the kernel dimensionalities are $5 \times 1 \times 1$, $4 \times 1 \times 1$, $5 \times 1 \times 1$, and $4 \times 1 \times 1$. The last two layers use a dropout mechanism to mitigate overfitting and improve model generalization. The dropout rate was set to 0.2. The number of activation maps is set to 16, 32, 64, and 128. We depict the cube-based variant in Figure 5.2.

As it can be seen, the model employs larger amount of feature maps, thus contains a higher number of trainable parameters and should provide a better unmixing performance. This variant is also concluded with two fully-connected layers. Furthermore, the authors in [278], experimentally proved the cube-based architecture's superiority over the pixel-based approach on unmixing the Urban benchmark dataset. The authors also compared this model to an auto-associative neural network (AANN) model proposed in [142], a linear mixing model (LMM) [88], and extended support vector machine (eSVM) introduced in [244]. The cube-based variant outperformed all of the baseline methods, including its spectral variation.

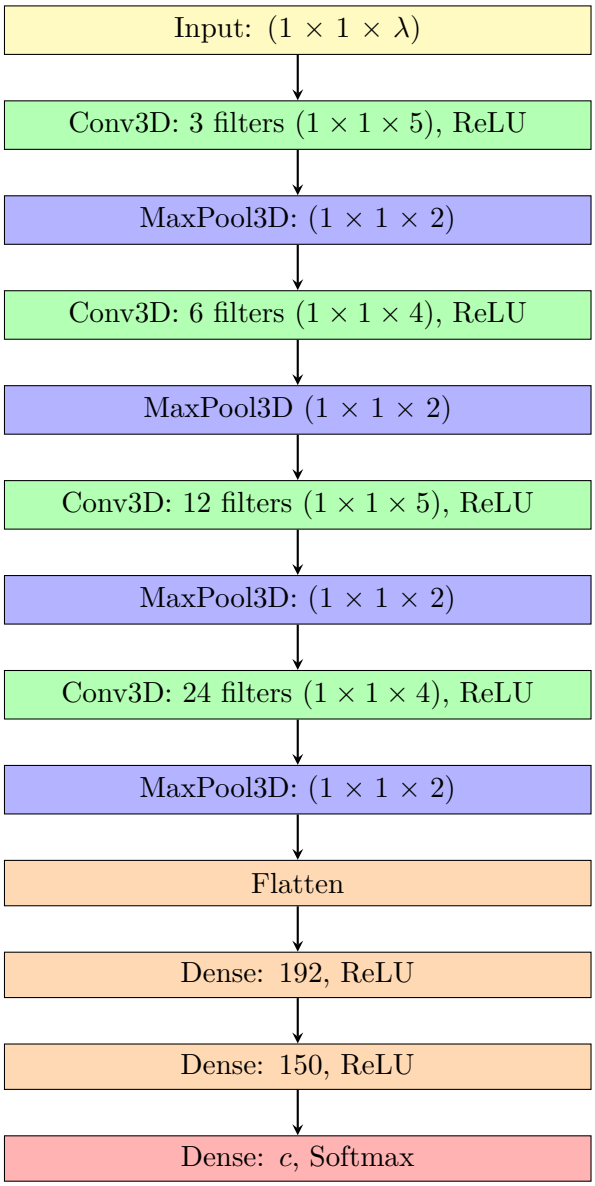


FIGURE 5.1: Pixel-based CNN architecture for hyperspectral unmixing [278]. Input shape is $1 \times 1 \times \lambda$, where λ is the number of spectral bands, and c is the number of endmembers in HSI.

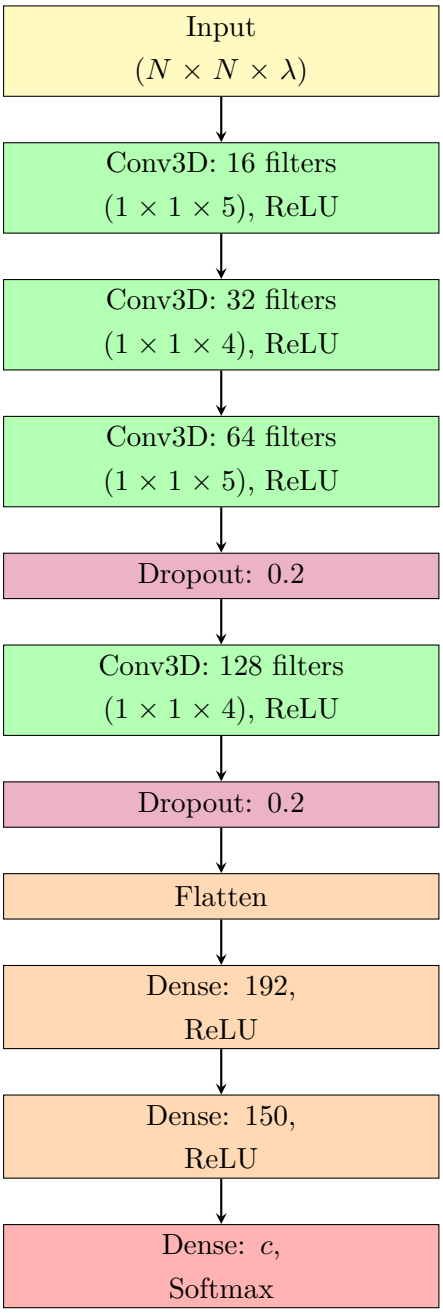


FIGURE 5.2: Cube-based CNN architecture for hyperspectral unmixing [278]. Input is $N \times N$ spatial neighborhood with λ spectral bands, output contains c abundances for each endmember in the HSI.

The second model, referred to as DCAE, was proposed in [112]. This model was the first attempt in HU to use an autoencoder-based architecture and utilize spectral-spatial information context. The objective function to train the model was set to spectral information divergence (SID) proposed in [29], coupled with the dropout technique. This model also incorporates two variants for spectral and spectral-spatial approaches respectively. The former one employs five convolutional layers where the kernel shape is set to $1 \times 1 \times 3$ and the number of feature maps follows 2, 4, 8, 16, and 32. This part of the model can be understood as a feature extractor,

which automatically learns different patterns to improve abundance estimation. All of the layers utilize ReLU non-linearity. The next part of the model incorporates two fully-connected layers which aim to learn how to properly construct the latent representation, which represents the abundance vector. The convolutional and fully-connected layers together form the encoder part of the model. The decoder includes a single fully-connected layer, where its weights are set to the endmembers' spectral matrix. Such operations allows the model to linearly combine the estimated latent representation with the constant matrix of the last layer (representing the endmembers spectra), to reconstruct the input sample. In this approach, the model learns how to estimate the unmixing vectors without explicitly utilizing the abundance ground-truth labels. Furthermore, a dropout technique is used in the first fully-connected layers in the encoder, with the rate set to 0.2. In the cube-based variant, the number of convolutional layers is set to four with its kernels dimensionality set to $3 \times 3 \times 3$, $3 \times 3 \times 3$, $1 \times 1 \times 3$, and $1 \times 1 \times 3$. All of the non-linearities are the ReLU functions and the number of activation maps are 16, 32, 64, and 128. Analogously to the spectral variant, the encoder part also incorporates the same fully-connected layers and the decoder constitutes a single matrix with its weights set to the endmembers spectra. In Figure 5.3, we provide visual representation of the pixel-based approach of the DCAE model, whereas in the Figure 5.4, we display topology of the cube-based variant. Finally, to showcase clear topology of the utilized models, we provide description in Tables 5.1 and 5.2, for both CNN and DCAE models, respectively.

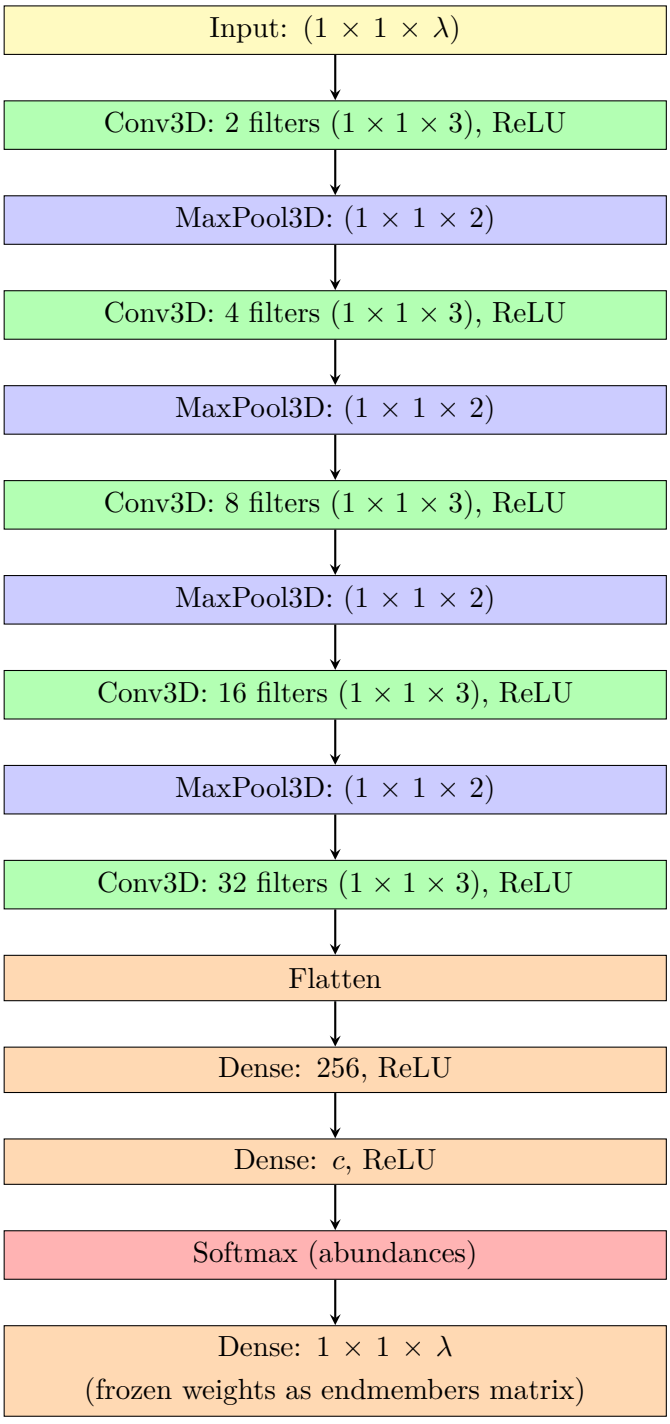


FIGURE 5.3: Autoencoder-based CNN architecture for hyperspectral unmixing proposed in [112]. Input is a 1×1 spatial pixel with λ spectral bands. The encoder learns abundances via convolutional and dense layers followed by a softmax, and the decoder reconstructs the spectrum using frozen endmember weights. Finally, c denotes the number of endmembers in the HSI.

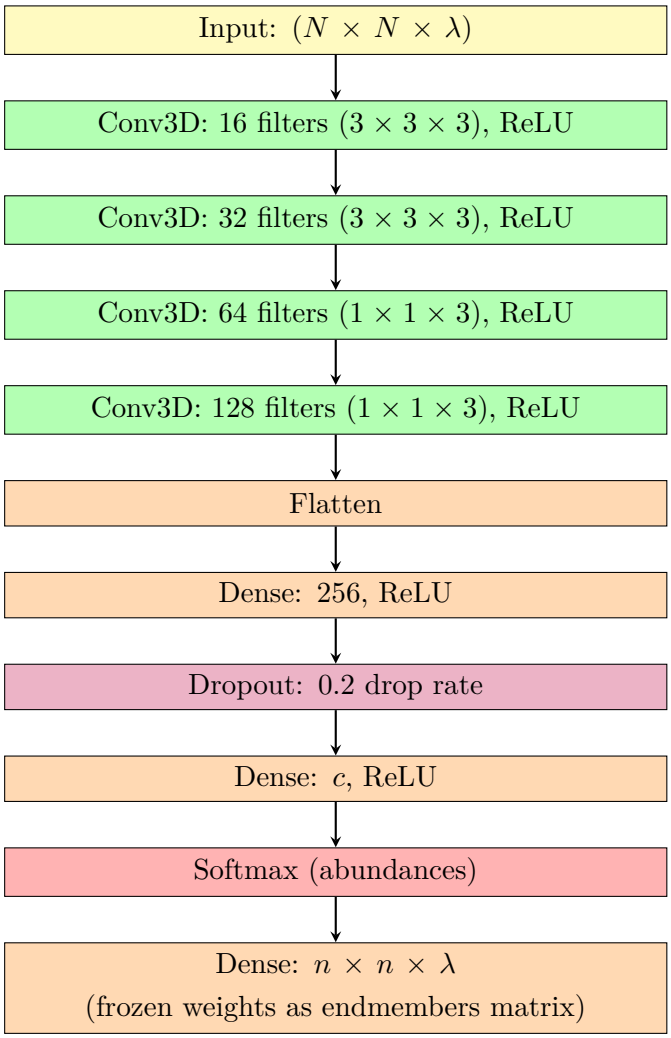


FIGURE 5.4: 3D DCAE model for hyperspectral unmixing [112]. Input is an $N \times N$ spatial neighborhood with λ spectral bands. The network extracts spectral-spatial features via 3D convolutions, learns abundance vectors (c) using dense layers, and reconstructs spectra using fixed endmember weights.

TABLE 5.1: The CNN architecture proposed in [278]. For each layer, we report the number of kernels, alongside their dimensions and activation functions. This table was included in our publication in [238].

Variant	Layer	Parameters	Activation
Pixel-based CNN $(1 \times 1 \times \lambda)$	Conv1	$3@1 \times 1 \times 5$	ReLU
	Conv2	$6@1 \times 1 \times 4$	ReLU
	Conv3	$12@1 \times 1 \times 5$	ReLU
	Conv4	$24@1 \times 1 \times 4$	ReLU
	FC1	$\# \times 192$	ReLU
	FC2	192×150	ReLU
	FC3	$150 \times a$	Softmax
Cube-based CNN $(3 \times 3 \times \lambda)$	Conv1	$16@1 \times 1 \times 5$	ReLU
	Conv2	$32@1 \times 1 \times 4$	ReLU
	Conv3	$64@1 \times 1 \times 5$	ReLU
	Conv4	$128@1 \times 1 \times 4$	ReLU
	FC1	$\# \times 192$	ReLU
	FC2	192×150	ReLU
	FC3	$150 \times a$	Softmax

TABLE 5.2: The DCAE architecture proposed in [112]. For each layer, we report the number of kernels, alongside their dimensions and activation functions. This table was included in our publication in [238].

Variant	Layer	Parameters	Activation
Pixel-based DCAE $(1 \times 1 \times \lambda) \rightarrow (1 \times 1 \times \lambda)$	Conv1	$2@1 \times 1 \times 3$	ReLU
	Conv2	$4@1 \times 1 \times 3$	ReLU
	Conv3	$8@1 \times 1 \times 3$	ReLU
	Conv4	$16@1 \times 1 \times 3$	ReLU
	Conv5	$32@1 \times 1 \times 3$	ReLU
	FC1	$\# \times 256$	ReLU
	FC2	$256 \times a$	+ Softmax
	FC3	$a \times \lambda$	ReLU
Cube-based DCAE $(5 \times 5 \times \lambda) \rightarrow (1 \times 1 \times \lambda)$	Conv1	$16@3 \times 3 \times 3$	ReLU
	Conv2	$32@3 \times 3 \times 3$	ReLU
	Conv3	$64@1 \times 1 \times 3$	ReLU
	Conv4	$128@1 \times 1 \times 3$	ReLU
	FC1	$\# \times 256$	ReLU
	FC2	$256 \times a$	+ Softmax
	FC3	$a \times \lambda$	ReLU

5.1.2 Experimental Settings

The neighborhood shape was set to 3×3 and 5×5 for the CNN [278] and DCAE [112] models, respectively. We train both models utilizing the Adam [116] optimizer with the batch size set to 256 to allow for decently-sized step during training and maintain stability. The learning rates for pixel- and cube-based CNNs were set to 0.001 and 0.0001, respectively, whereas the learning rates for DCAE for spectral and spectral-spatial variants were equal to 0.00001 and 0.00005. The maximum number of training epochs was set to 100 with an early-stopping mechanism conditioned on the validation loss equal to 15 epochs without improvement.

5.1.3 Benchmark Datasets

In this study, we focus on two HU well-established benchmarks, i.e., the Jasper Ridge and Urban datasets. The first one originally employs 512×614 pixels, where each one captures 224 channels. The electromagnetic spectrum spans from 380 nm to 2500 nm. This HSI is further cropped in to a 100×100 window with four underlying endmembers: *i*) road, *ii*) water, *iii*) soil and *iv*) Tree. After removing contaminated and water vapor channels, the HSI incorporates 198 bands. In the Urban benchmark, the HSI spatial shape is 307×307 and the number of bands is equal to 162. This benchmark comprises six endmembers: *i*) Asphalt, *ii*) Grass, *iii*) Tree, *iv*) Roof, *v*) Metal, and *vi*) Dirt. As it can be seen, both benchmarks provide different complexity in terms of spectral or spatial domains, i.e., the Jasper Ridge contains a higher number of bands, thus making it more complex to interpret in this domain, whereas the Urban benchmark uses more pixels, hence introducing more context in these axes. Such selection of benchmarks allowed us to ensure variability and provide fair and representative comparison between the investigated methods as well as the impact of varying size of the training set.

Note: In this thesis, we give the description of utilized datasets potentially multiple times, in order to make the corresponding sections self-contained. Although it would be possible to keep them in a single place in the thesis, we believe that referring to a fairly "distant" section would negatively impact the read.

In all experiments, we incorporate a cross-validation strategy with Monte-Carlo technique and sample 30 variations of training ($\boldsymbol{T}$) and test sets (Ψ). It is critical to emphasize that the test set elements do not change with the change of the size of the training set size. The test size for the Jasper Ridge benchmark is set to 7500, whereas for Urban dataset it is equal to 47000 pixels. We evaluate the impact of the size of the training set by investigating the following variants {1%, 6%, 13%, 33%, 66%} of randomly sampled pixels from the original benchmark HSIs. Thus, for Jasper Ridge and Urban datasets we constructed the sets with {75, 500, 1000, 2500, 5000} and {470, 2800, 6100, 15500, 31000} elements. The validation set incorporates 10% of the available training pixels for both benchmarks and utilized for the early stopping

condition mechanism. Naturally, such pixels are removed from training set to remove duplication.

5.1.4 Quality Metrics

To evaluate the HU capabilities of investigated models, we incorporate two metrics that measure the distance between the estimated and ground-truth abundance vectors. The first one is the RMSE, whereas the second metric is rmsAAD. RMSE can be formulated as:

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{N}(\boldsymbol{a}_i - \hat{\boldsymbol{a}}_i)^2}{N}}, \tag{5.1}$$

whereas the rmsAAD can be given as:

$$\text{rmsAAD} = \sqrt{\frac{\sum_{i=1}^{N} \arccos(\frac{\boldsymbol{a}_i^\top \hat{\boldsymbol{a}}_i}{\|\boldsymbol{a}_i\|\|\hat{\boldsymbol{a}}_i\|})^2}{N}}. \tag{5.2}$$

The N is the number of samples in the test set that remains constant with the change of size of the training sets, $\boldsymbol{a}$ and $\hat{\boldsymbol{a}}$ are the ground-truth and estimated abundance vectors. **Note**: In this thesis, we give the formulas for the most important quantitative metrics potentially multiple times, in order to make the corresponding sections self-contained. Although it would be possible to keep them in a single place in the thesis, we believe that referring to a fairly "distant" section would negatively impact the read.

5.1.5 Experimental Results and Discussion

In Figures 5.5 and 5.6, we provide the RMSE and rmsAAD values estimated for each variant of the models (CNN and DCAE), coupled with the varying sizes of the training set (on the horizontal axis). To perform statistical analysis and tests, we filter out the outlier experimental runs employing the inter-quartile range (IQR). This procedure allowed us to calculate the mean and median statistics without the influence of outlying runs. More specifically, we retain executions for which RMSE is in between $\mathrm{Q1} - 1.5 \cdot \mathrm{IQR}$ and $\mathrm{Q3} + 1.5 \cdot \mathrm{IQR}$, where Q1 and Q3 are the first and the third quartile. For both models, there are approximately two such outlying experiment runs per variant.

For the DCAE model, similarly as in the original work [112], the cube-based model outperforms the pixel-based variant, which shows that spatial context information is effectively employed in the latent representation elaborated by DCAE. Furthermore, it implies that in the neighborhood data, there are important explanatory features, which allow the model to enhance the estimated abundance vectors. Such phenomenon was not manifested in CNNs variants. In the experimental results, the pixel-based model obtained lower regression errors when compared to the cube-based CNN. This can be attributed to noise inherently present in the spatial dimensions that negatively influenced the quality of unmixing predictions. Moreover, if the ground sampling distance is too large, some fine-grainted features could be difficult to recognized by the cube-based variant. Subsequently, as HU is a multitarget regression problem, the models are more sensitive to fluctuations in the input data when compared to e.g., classification or segmentation models.

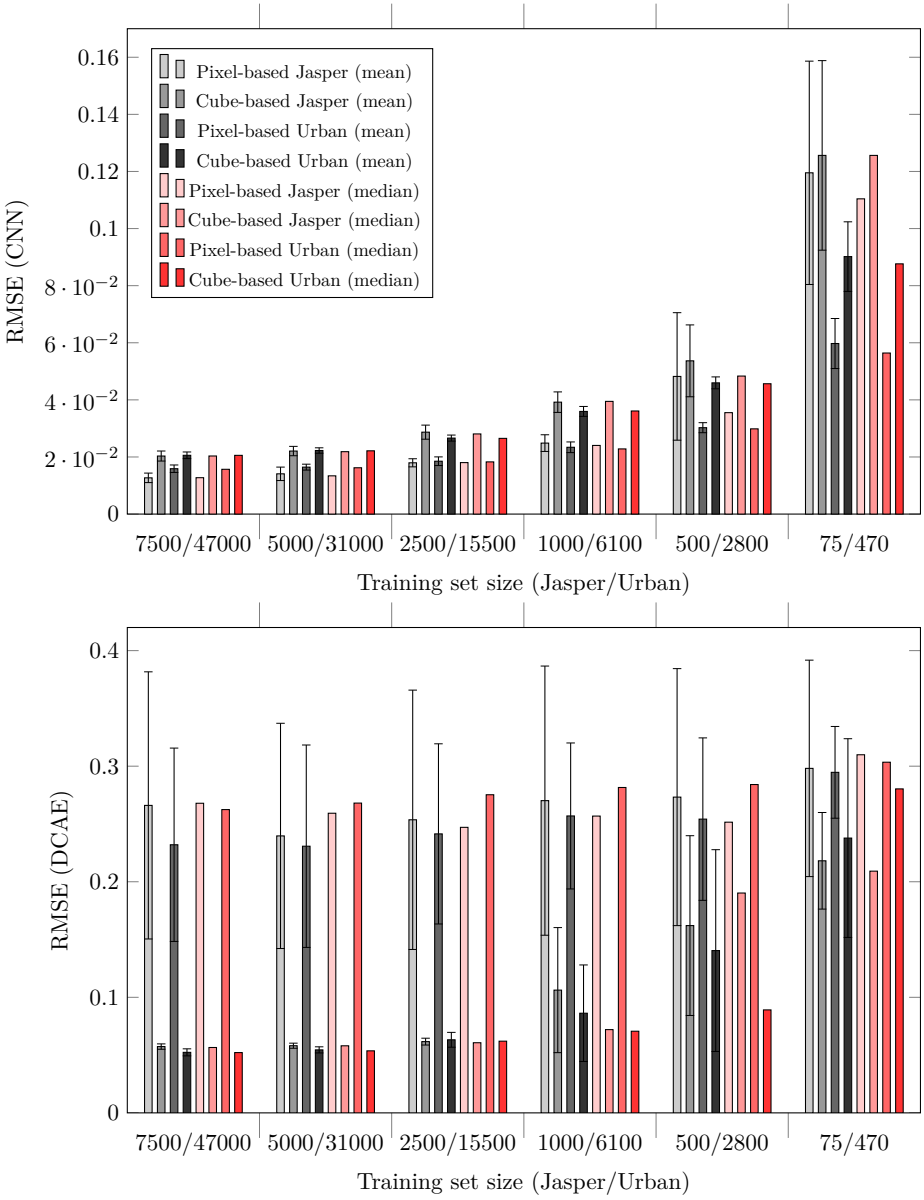


FIGURE 5.5: The experimental results for RMSE metric. We provide values for the pixel- and cube-based configurations of CNN and DCAE models. The averages are rendered in gray, whereas the medians are presented in red (the whiskers indicate standard deviation). This figure was included in our publication in [238].

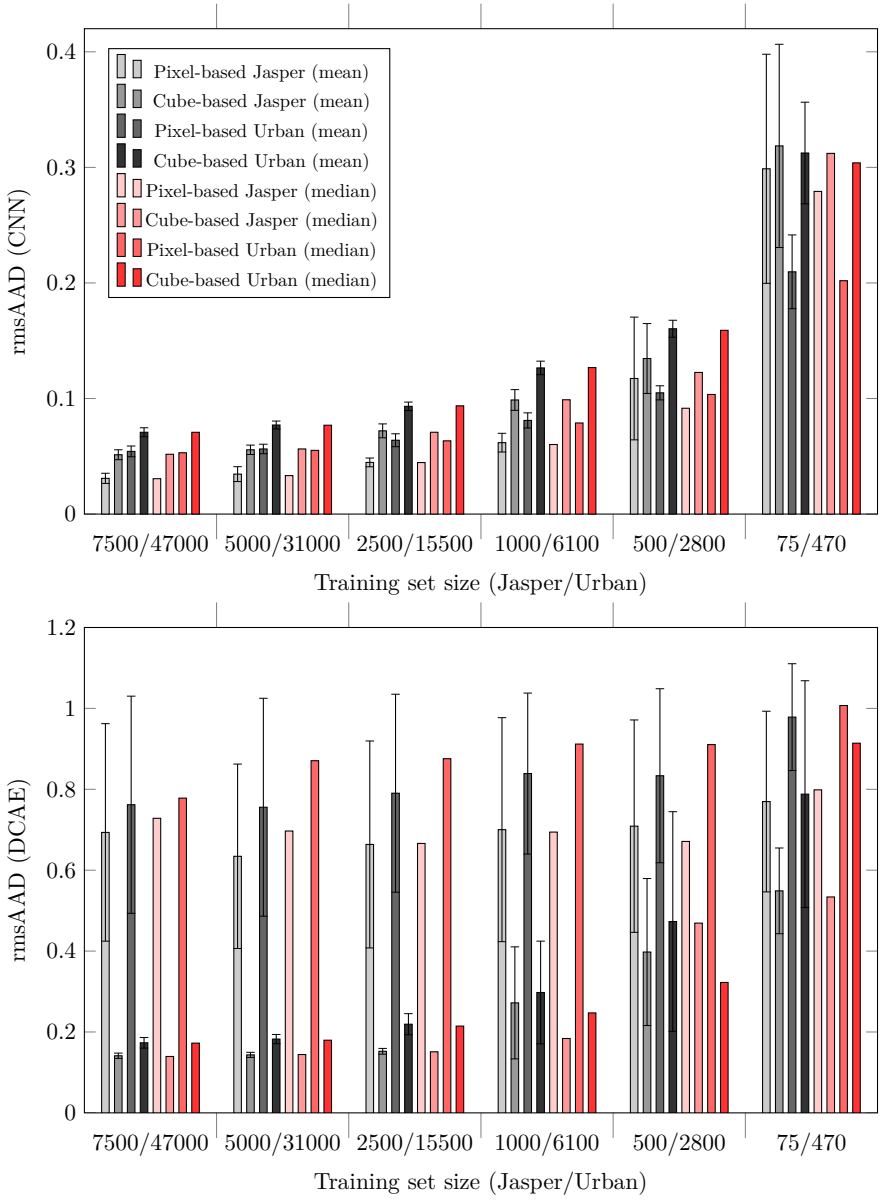


FIGURE 5.6: The experimental results for rmsAAD metric. We provide values for the pixel- and cube-based configurations of CNN and DCAE models. The averages are rendered in gray, whereas the medians are presented in red (the whiskers indicate standard deviation). This figure was included in our publication in [238].

TABLE 5.3: The metrics of the Friedman tests followed by the Dunn's multiple comparisons tests for (a) Jasper Ridge as well as (b) Urban benchmarks calculated for rmsAAD. We highlight pairs (a–b, where a and b are the percentage of all training pixels, in %) for which the averages are statistically significantly different (at $p < 0.05$). This table was also included in the PhD Candidate's publication in [238].

Variant	Set	Pairs
Pixel-based CNN	(a)	1–13, 1–33, 1–66, 1-100, 6–33, 6–66, 6–100, 13–66, 13–100
	(b)	1–33, 1–66, 1–100, 6–66, 6–100, 13–100
Cube-based CNN	(a)	1–13, 1–33, 1–66, 1–100, 6–33, 6–66, 6–100, 13–66, 13–100
	(b)	1–33, 1–66, 1–100, 6–66, 6–100, 13–66, 13–100
Pixel-based DCAE	(a)	—
	(b)	1–33, 1–66, 1–100
Cube-based DCAE	(a)	1–13, 1–33, 1–66, 1–100, 6–66, 6–100
	(b)	1–100

In this study, the values show that decreasing the size of the training sets severely affects the unmixing error (Table 5.3). To evaluate the statistical relevance of these results, we performed Friedman tests in conjunction with Dunn's multiple comparison procedures for each dataset. For brevity, we focus on rmsAAD, since the outcomes for RMSE correspond and correlate closely with those of rmsAAD. The examination of this metric indicates that extremely small training sets (comprising 1% and 6% of the overall data) lead to significantly inferior unmixing quality in contrast to models trained on more extensive datasets (spanning from 33% to 100% of the data). Importantly, when utilizing adequately sized reduced training sets (33% of the available data), no substantial decrease in unmixing efficacy was noted for either model architecture, even when juxtaposed with training on larger datasets (surpassing 66% of the data). This implies that increasing the quantity of manually annotated training pixels may not considerably improve the models' performance under these conditions.

5.1.6 Concluding Remarks

The experimental results obtained over two benchmarks indicated that the deterioration of the abundance estimation quality is especially present for smaller training sets, i.e., up to 6% of the HSI. Furthermore, when increasing the size of the training sets above the threshold of 33% did not bring statistically significant improvements. The experimental results show that the manual labeling process should be coupled with the quantitative and statistical analysis to determine the required and sufficient number of training samples.

We utilized the cube-based variants of both models without asserting the information leak that may be present when the neighboring pixels of training and test sets overlap with each other. Such phenomenon may lead to overly-optimistic results and must be studied carefully. In the subsequent Section 5.2, we investigate another sampling strategy to mitigate the information leak and expanding this study. Such issue is critical when it comes to model evaluation for multifold reasons, with the

pivotal ones being operational and deployment safety, model generalization capabilities, inference expected results, and proactive root cause error handling—by fair and unbiased quantitative evaluation. Finally, in Section 5.2, we continue the research on the influence of the varying training set sizes.

5.2 Unbiased Validation and Representative Sampling Strategy of Hyperspectral Unmixing Algorithms

The initial step in our framework concerns establishing a representative and fair division of an input hyperspectral scene into the training and test subsets. As already mentioned, the evaluation of supervised HU methods is significantly influenced by the limited availability of GT data. Typically, the hyperspectral pixels used for the training set (T) and the test set (Ψ) are extracted from the same HSI, which represents a **singular scene**. Such pixels are spatially correlated and captured under identical acquisition conditions. Because of this phenomenon, it disables the possibility of replicating the operational scenarios encountered by Earth observation satellites in real-world applications. In HU, this challenge is more visible, as satellite images possess significantly lower spatial resolution compared to airborne HSIs, where the spectral features are of better fidelity.

Even in cases where multiple HSIs are accessible, they often encompass different endmembers. In such situations, it is extremely difficult (or even impossible) to train using one HSI and test with another since the targets are completely different. While deriving T and Ψ from the same HSI is considered valid, ensuring there is no leakage of information between these datasets is essential [109], [176], especially for sensors with low spatial resolution. For methods focused solely on spectral features, it suffices to prevent identical pixels from being present in both subsets. However, in spectral-spatial analysis when considering 3D samples from the HSIs, it is necessary to exclude pixel spatial neighborhoods from training to avoid overlaps. Such situation renders when the same pixel appears within the neighborhoods of different pixels in both training and testing sets, causing the leak of information across those subsets. Moreover, when training or evaluating models on such data, the results will most likely be over-optimistic, hindering the ability to draw sensible conclusions or performing statistical analysis of such data [176]. Furthermore, this issue has been examined in the literature in the context of HSI classification in [176]. Many of HU studies tend to overlook this significant limitation and discard the neccessity to split such sets properly, asserting no information leak. Moreover, as highlighted in [7], [23], HSIs are generally smaller in terms of number of pixels when compared to datasets in other imaging modalities, making this problem even more pressing.

5.2.1 Extracting Training-Test Dataset Splits

To extract the T and Ψ division, we propose a multistep approach (for visual example, please refer to Figure 2.1) called *Geospatially Stratified Validation* (GSV) that given an input HSI image of height H and width W as well as the expected size of T and Ψ ($|T|$ and $|\Psi|$ respectively) performs the following:

1. Sample randomly the *start* index based on the maximum H and W variables:

   $h_s, w_s \leftarrow randint(H), randint(W)$,

   where *randint* operation samples integers from the discrete uniform distribution in the half-open interval $[low, high)$. In this case, the lower and upper bounds are 0 and H or W, respectively.

2. Calculate the flattened index from the 2D image by treating it as it was a 1D projection via concatenating rows sequentially:

   $i_e \leftarrow h_s W + w_s$.

3. Calculate the *end* index given the size of the training set $|T|$:

   $i_e \leftarrow i_s + |T|$.

4. Given the start and end indices for the training set, we have two possibilities: *i*) the i_e index is larger than i_s—this happens when the end did not wrap around the HSI, and *ii*) the opposite case of $i_e < i_s$. The situation when $i_s = i_e$ is logically not possible for a real example, thus omitted by our algorithm. This can happen when the magnitude of the training set T is set to be equal the number of pixels in the HSI. In the former example of $i_e < i_s$, the end index did overshoot the maximum dimensions thus was reduced by the modulo operator and wrapped around the image.

   In the first case for $i_e > i_s$, we incorporate all pixels between start and end indices to the training set T. Subsequently, for the test set Ψ, we populate it with all the remaining pixels, i.e., from i_e up to the end of the image i_{H+W}, as well as from the first pixel in the 1D array i_1 up to i_s.

   In the opposite case, when $i_e < i_s$, we employ a reverse strategy, i.e., the T set is populated with pixels from i_s up to i_{H+W} and i_1 to i_e. The test set is filled with the remaining samples between i_e and i_s (since $s > e$).

   To obtain specific spatial 2D indices of each index, e.g., index i_k, the required formula is the following:

   $h_k \leftarrow \left\lceil \frac{i_k}{W} \right\rceil \mod H$,

   for the height index and

   $w_k \leftarrow i_k \mod W$, for the width index.

5. Finally, since the magnitude of the test set was specified, we additionally subsample Ψ to a required size $|\Psi|$.

In our algorithm, the final test set is redefined to only include a subset of its elements from the existing set Ψ. Specifically, our algorithm takes the last $|\Psi|$ pixels (or patches) from the total number of elements in Ψ and counts backward to select only those elements that fit within the specified size.

The purpose of this step is to ensure that the test set conforms to a predefined size limit, which can be crucial for various applications such has DL models validation. Another important advantage is that if the length of the training set $|T|$ is increased, such backward sampling allows to maintain the exact same elements in Ψ. It is critical during various types of experiments, when the test set must remain constant (with the change of $|T|$).

The specified algorithm can be utilized for spectral and spatial-spectral approaches, where in the latter case, each index does not only represent a specific pixel, but a local neighborhood of pixels. To prevent from the information leak phenomenon, it is crucial to only extract non-overlapping patches. In the case of edge pixels—which normally would be included in the neighboring regions in both sets (T and Ψ), thus introducing information leak—we do not incorporate them in any set and are discarded from analysis. For a visualization of such procedure please refer to Figure 2.1.

Finally, although using our proposed algorithm allows to effectively alleviate and remove the problem of information leak, nonetheless, given the fact that T and Ψ are from the same HSI, the samples from both sets are likely spatially correlated and could share the same endmembers that are inherently embedded in the imagery. Furthermore, the samples from training and test sets were acquired with the same hyperspectral sensor, sharing the same acquisition parameters and effects such as noise related to the atmosphere or the camera itself. To completely remove such correlations from those sets, it is necessary to examine HSIs from different regions of interest and preferably change sensor characteristics to completely validate robustness of validated models.

There are numerous benefits of this algorithms with the main contribution being:

- Operating only on indices allows us for fast and memory efficient sampling and division on training and test sets.
- The size of T and Ψ are adjustable.
- The test set remains constant with the changing size of training set. Hence, such training-test splits might be used to validate the robustness of machine learning models against different sizes of the training sets, and this analysis would be performed on the very same test set. This, is turn, allows us to directly confront the obtained quality metrics across different models, and ensures full reproducibility of such experimentation.
- Our algorithm removes the risk of spatial information leak and allows us to sample non-overlapping neighboring regions between pixels from T and Ψ.

- Our approach strives in its simplicity, thus allowing for easy modification, offering high flexibility and versatility.
- The proposed sampling strategy can operate on any image—not only on HSIs but also could be used for semantic or instance segmentation of even red, green, blue (RGB) images.
- We can extract multiple training-test folds to simulate cross validation scenarios, via sliding window operation across the input image.
- Our algorithm can be used for pixel- and patch-based approaches.

5.2.2 Experimental Settings

In the experiments, we evaluate the influence of T and Ψ sampling process and compare performance of state-of-the-art methods for HU. We compare a standard and universally used random sampling (RS) approach with our proposed—GSV—sampling algorithm. It is worth emphasizing that the stratification approach widely used in the literature does not incorporate any mechanism to detect and alleviate the problem of spatial information leak, thus such scenario is highly probable. Hence, may lead to overoptimistic results when benchmarking models and provide inconclusive results. Furthermore, RS method does not prevent from injecting spatially-correlated neighborhood across training and test subsets.

Our experiments were conducted utilizing *Python 3.6* and the entire implementation is freely accessible under the link `https://gitlab.com/jnalepa/hu_validation.git`. The selected state-of-the-art models were trained with the same hyperparameters and we used the Adam optimizer with the learning rate set to 0.001. Moreover, the maximum number of epochs to train DL models was set to 100, where the early stopping condition—that allows us to stop the training before reaching the maximum number of epochs if the validation loss is not decreasing—was 15 epochs. In our experiments the validation set incorporates 10% of T, consequently those elements are removed from the training set. Furthermore, the batch size was set to 256, hence allowing the Adam optimizer to incorporate a sufficiently large step to avoid overfitting, oscillation and allow for smooth convergence. For the loss function that is utilized by the optimizer, we utilize mean square error.

In the study, we evaluate different models, ranging from classical method used in HU to DL models that manifest the recent advancements in this field. The first technique that we evaluate is a LMM approach proposed in [88]. It is a fully constrained least squares method with two constraints incorporated: *i*) the abundance nonnegativity constraint, and *ii*) the abundance sum constraint. The next classical model selected as baseline was a Support Vector Regression (SVR) model with one instance per target to fit the multitarget regression HU task presented in [33].

Moreover, we evaluate a CNN model adjusted for the unmixing task proposed in [278]. In our experiments we select the more advanced cube-based variant of this

architecture, by using an additional local neighborhood to improve the unmixing performance. This model incorporates four 3D convolutional layers as feature extractor segment and is concluded with three fully-connected layers with additional dropout mechanism to avoid overfitting. The kernel dimensions are $1 \times 1 \times 5$ and $1 \times 1 \times 4$ with unit stride and are used consecutively, one after another. In the regressor block which exploits fully-connected layers, the number of units is 192, 150 and C, where C denotes the number of endmembers in the HSI. For the activation functions, the authors utilized the ReLU non-linearity for all but the last layers, which incorporates the Softmax function [12] to satisfy the abundance constraints. We refer to this model as CB-CNN in the experiments and provide visualization of its architecture in the Figure 5.7.

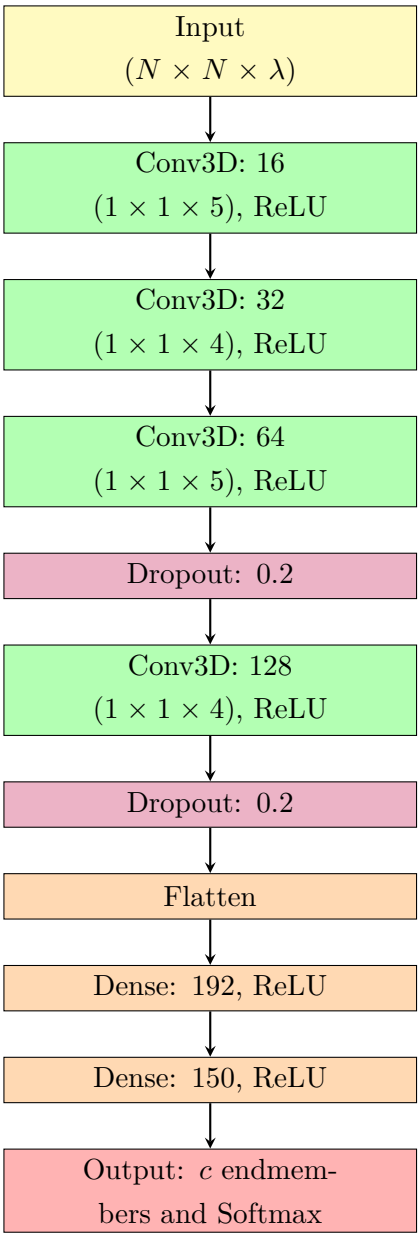


FIGURE 5.7: Cube-based CNN (CB-CNN) architecture for hyperspectral unmixing [278]. N defines the spatial size of the input sample, whereas λ represents the number of bands in the HSI. The number of endmembers is referred to as c.

Finally, we select an additional model from the literature proposed in [10]. This research introduces a weakly supervised deep CNN model to tackle the task of HU. The proposed model architecture resembles an autoencoder and it can be decomposed into the encoder and decoder parts. The former one incorporates a feature extraction block that is composed of 5 Network blocks and a linear layer followed by the Softmax non-linearity to obtain the abundance maps as the latent representation. Each Network block encapsulates a convolutional layer with $2 \times 2 \times 7$ initial kernel and following ones of dimensionality $2 \times 2 \times 5$ with ReLU activation function. The decoder utilizes the estimated abundance maps and reconstructs the input sample.

In the Figure 5.8, we provide visual representation of the utilized architecture. This model can also work in spectral and spectral-spatial fashion, hence in our experiments we selected the more advanced cube-based version. We refer to this method as weekly-supervised autoencoder (WS-AE).

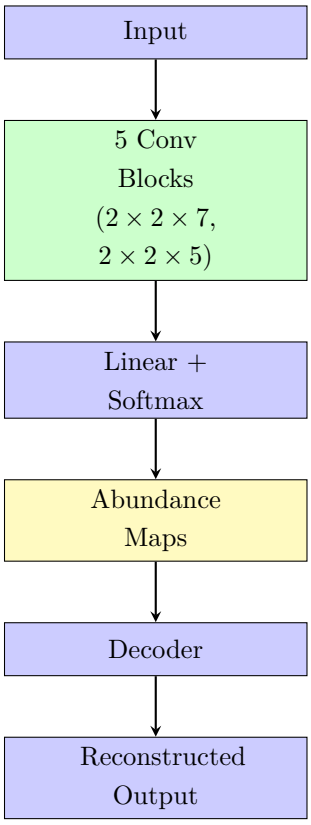


FIGURE 5.8: WS-AE (Weakly-Supervised Autoencoder) architecture for hyperspectral unmixing [10]. We highlight the 5 convolutional blocks with feature reduction to obtain the estimated abundance vectors. The decoder is set to the endmembers matrix and is utilized to reconstruct the input sample via combination of abundances and its weights.

5.2.3 Benchmark Datasets

In the experiments we selected the most well-known and established unmixing benchmarks, i.e., the Samson (Sa) with spectral-spatial dimensionality of $95 \times 95 \times 156$, hence incorporating 156 bands. It encapsulates three endmembers: *i*) Soil, *ii*) Tree, and *iii*) Water. The subsequent dataset is Urban (Ur) containing 307×307 pixels accompanied with 162 bands. This benchmark incorporates six endmembers: *i*) Asphalt, *ii*) Grass, *iii*) Tree, *iv*) Roof, *v*) Metal, and *vi*) Dirt. Finally, the last dataset utilized for evaluation is Jasper Ridge (JR) with 100×100 spatial shape and 198 bands. The following four endmembers are present in this scene: *i*) Road, *ii*) Water, *iii*) Soil, and *iv*) Tree.

To perform statistical analysis of the results, each experiment is repeated 30 times following a 30-fold Monte Carlo cross-validation scheme. Utilizing RS and our GSV

approaches, we sample 30 test sets that do not change size with the change of T length. The size of $\mathbf{\Psi}$ for Sa, Ur and JR is set to 3025, 47249, and 2500 respectively. The total size of the training set is set to 6000, 47000 and 7500 for Sa, Ur, and JR.

We also investigate the influence of varying training set size on the quality of unmixing in the context of RS (sampling with information leak) and GSV (stratification without information leak). This part of experiments is crucial to measure how different sampling techniques cope with limited training data and what is the consequence of training on such samples. Furthermore, it allows us to understand the relationship between the presence of information leak and varying training set sizes. We incorporate such magnitudes for $\boldsymbol{T}$ compared to full HSI in such percentages: $1\%, 6\%, 13\%, 33\%, 66\%$. Subsequently, such sampling creates the following training set sizes for Sa, Ur and JR:

- Sa—$60, 360, 780, 1980, 3960$
- Ur—$470, 2800, 6100, 15500, 31000$
- JR—$75, 500, 1000, 2500, 5000$

5.2.4 Quality Metrics

To evaluate the quality of unmixing we utilize two metrics: *i*) RMSE and *ii*) rmsAAD. The formula for RMSE can be denoted as:

$$\text{RMSE} = \sqrt{\sum_{i=1}^{|\Psi|}(a_i - \hat{a}_i)^2/|\Psi|}, \tag{5.3}$$

whereas rmsAAD can be described using the following formula:

$$\text{rmsAAD} = \sqrt{\sum_{i=1}^{|\Psi|}\arccos\left(\frac{a_i^\top \hat{a}_i}{\|a_i\|\|\hat{a}_i\|}\right)^2/|\Psi|}, \tag{5.4}$$

where $|\Psi|$ is the size of the test set Ψ, and $\boldsymbol{a}$ and $\hat{\boldsymbol{a}}$ are the ground-truth and predicted abundances. It is worth emphasizing that both metrics should be minimized and the perfect score is equal to zero.

5.2.5 Experimental Results and Discussion

In Table 5.4 we present RMSE and rmsAAD for both sampling strategies, whereas in Table 5.5, we show the rankings for both metrics, calculated with all models averaged across all benchmark datasets. From the experimental results we can observe that the CB-CNN architecture outperforms other methods on all benchmark datasets, including the patch-based WS-AE model. If the results are averaged for all variations of training set sizes, we also appreciate the best performance in unmixing for the CB-CNN model. It is also worth emphasizing that utilizing the spectral-spatial features is most beneficent and allows the model to improve its precision in estimating

the abundances over $\boldsymbol{\Psi}$ by exploiting more informative context in the feature maps. Moreover, the experimental results also convey that increasing the size of $\boldsymbol{T}$ leads to improvement in unmixing over the test set, which should be an expected scenario. Furthermore, the classical techniques such as LMM failed short when compared to more advanced and recent DL architectures in the quality of abundance estimation over Sa and Ur. In the case of JR, LMM provided better results than CB-CNN and WS-AE. Such phenomenon is probably caused by a very limited number of training samples, thus bounding the learning and generalization capabilities of DL models. Such limitation emphasizes a real problem of the sparse amount of labeled data in HU and could be resolved via various augmentation techniques at a cost of longer training times and increased compute resources.

We evaluated the impact of sampling strategy on the unmixing capabilities of all methods. We investigated the metrics obtained for both RS and GSV for all Monte-Carlo 30 cross-validation folds. It is important to emphasize that the test set $\boldsymbol{\Psi}$ was kept constant and was the same for RS as well as for GSV. In the case of training sets $\boldsymbol{T}$, the elements were sampled differently as both techniques incorporate divergent strategies. We calculate the delta for both metrics, i.e., RMSE:

$$\delta_{RMSE} = (RMSE_{GSV} - RMSE_{RS})/RMSE_{RS}, \tag{5.5}$$

and rmsAAD:

$$\delta_{rmsAAD} = (rmsAAD_{GSV} - rmsAAD_{RS})/rmsAAD_{RS}. \tag{5.6}$$

Consequently, δ is the real difference between the GSV and RS sampling methods. Furthermore, if the information leak which is present in RS would have no effect on the HU metrics, the delta would be equal to zero. Values > 0 indicate that for GSV the unmixing error was larger than for RS. Thus RS division would in effect render overly-optimistic results of abundance estimation, which is a critical point in this research. In Table 5.6, we present the deltas calculated for both metrics with varying size of $\boldsymbol{T}$. From the results it is clear that for all cases the RS approach produces lower errors for both metrics, hence produces overly optimistic results for all benchmarks. The GSV errors could even be up to > 5 times larger than results for RS. It is clear that RS does not allow to obtain a fair stratification between training and test sets, and incorporates information leak, which in result, causes the experimental results to be overly optimistic and not conclusive. Furthermore, such pattern in the HU metrics is also visible in Figure 5.9 and 5.10, where we depict RMSE and rmsAAD values via a radar plot visualization for both metrics, respectively. The both metrics are much closer to the origin for RS approach. What is worth of observing is that the rankings calculated in Table 5.5 are positively correlated for both sampling strategies, thus both stratification methods could be utilized for model selection. Nonetheless, the information leak will cause the RS values to be overly optimistic, hence its metrics would not be usable in further analysis or statistical testing. What is more, the overly

optimistic values of RS could lead to operational problems and massive failures if any model validated in this paradigm would be deployed in real use cases, e.g., satellite sensors. In such conditions, the produced results would be of lesser quality and could potentially be the root cause of many operational issues.

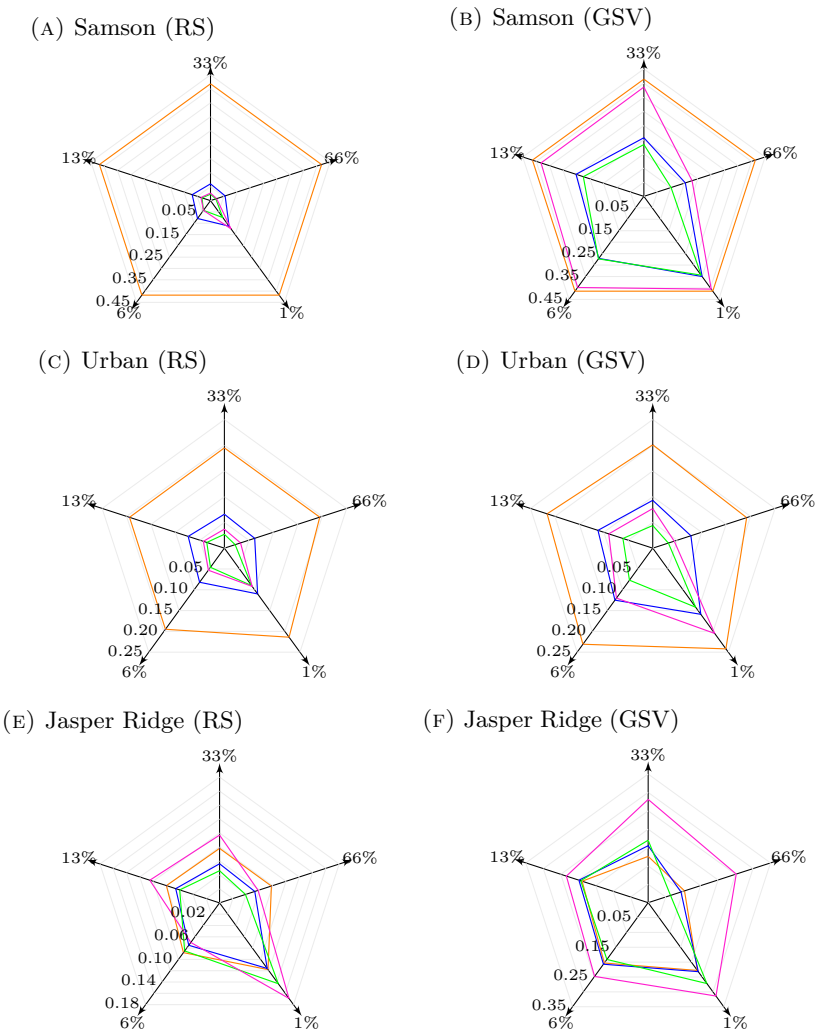


FIGURE 5.9: The RMSE values for both sampling strategies (RS and GSV) and all datasets: LMM [88], SVR [34] CB-CNN [280], WS-AE [11]. The values closer to zero indicate better HU quality.

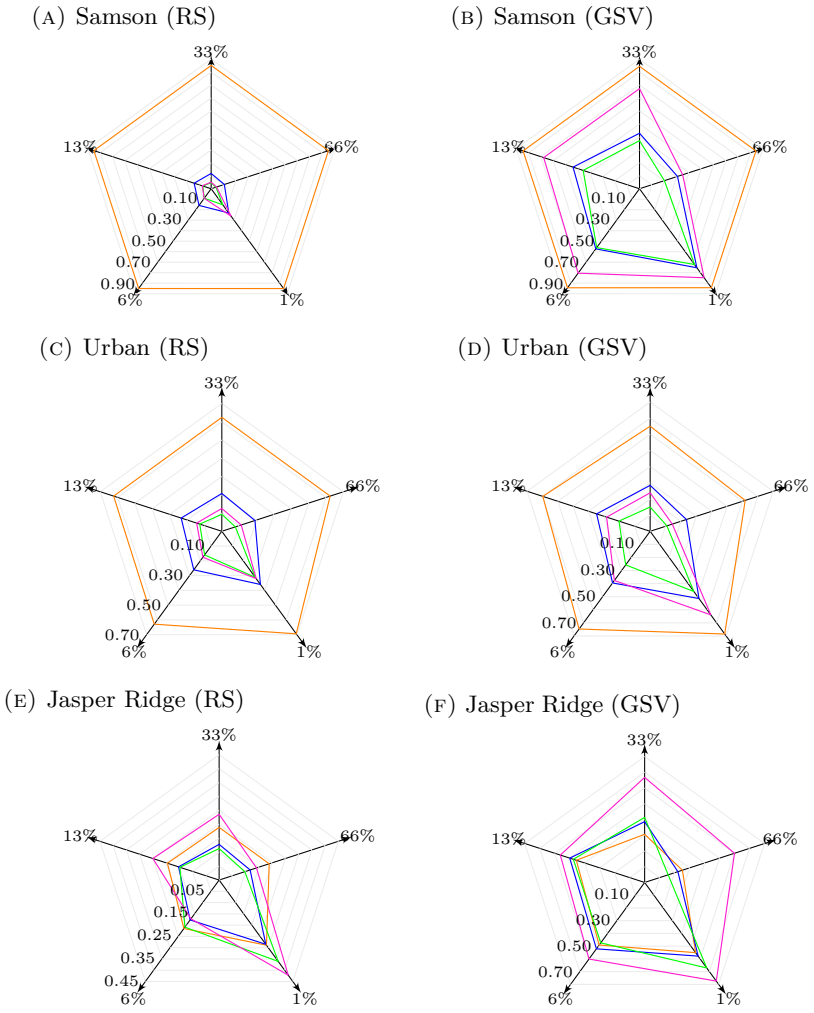


FIGURE 5.10: The rmsAAD values for both sampling strategies (RS and GSV) and all datasets: ■ LMM [88], ■ SVR [34] ■ CB-CNN [280], ■ WS-AE [11]. The values closer to zero indicate better HU quality.

TABLE 5.4: RMSE and rmsAAD measured for the investigated HU methods and sampling strategies (RS and GSV), averaged across all benchmarks. The best results for each sampling strategy and training set size are boldfaced, and the second best—underlined. This table was also utilized in the PhD Candidate's work in [235].

	Train. set size →	**66%**		**33%**		**13%**		**6%**		**1%**		**Mean**	
	Data split →	**RS**	**GSV**	**RS**	**GSV**	**RS**	**GSV**	**RS**	**GSV**	**RS**	**GSV**	**RS**	**GSV**
RMSE	LMM [88]	0.230	0.237	0.230	0.247	0.231	0.273	0.234	0.283	0.250	0.295	0.235	0.267
	SVR [34]	0.057	0.109	0.061	0.152	0.070	0.187	0.079	0.201	0.113	**0.247**	0.076	0.179
	CB-CNN [280]	**0.027**	**0.066**	**0.032**	**0.132**	**0.043**	**0.160**	0.058	**0.180**	**0.103**	0.253	**0.053**	**0.158**
	WS-AE [11]	0.038	0.158	0.054	0.248	0.061	0.235	**0.055**	0.255	0.129	0.308	0.067	0.241
rmsAAD	LMM [88]	0.588	0.605	0.588	0.633	0.590	0.699	0.598	0.727	0.645	0.760	0.602	0.685
	SVR [34]	0.138	0.255	0.151	0.366	0.175	0.462	0.199	0.496	0.292	**0.615**	0.191	0.439
	CB-CNN [280]	**0.071**	**0.156**	**0.084**	**0.311**	**0.114**	**0.379**	0.153	**0.431**	**0.276**	0.618	**0.140**	**0.379**
	WS-AE [11]	0.099	0.365	0.137	0.559	0.156	0.541	**0.146**	0.594	0.335	0.754	0.175	0.562

TABLE 5.5: The ranking for both investigated metrics, i.e., RMSE and rmsAAD for the evaluated methods and different sampling strategies (RS and GSV), averaged across all benchmarks. The best scores for each training set size and sampling strategy are boldfaced, and the second best are underlined. This table was also utilized in the PhD Candidate's work in [235].

	Train. set size →	**66%**		**33%**		**13%**		**6%**		**1%**		**Mean**	
	Data split →	**RS**	**GSV**	**RS**	**GSV**	**RS**	**GSV**	**RS**	**GSV**	**RS**	**GSV**	**RS**	**GSV**
RMSE	LMM [88]	4.000	3.667	3.667	3.000	3.667	3.000	4.000	3.333	3.333	3.000	3.733	3.200
	SVR [34]	2.667	2.333	2.667	2.333	2.667	2.667	2.667	2.333	2.000	2.000	2.533	2.333
	CB-CNN [280]	**1.000**	**1.000**	**1.000**	**1.667**	**1.000**	**1.333**	**1.667**	**1.333**	**1.667**	**1.667**	**1.267**	**1.400**
	WS-AE [11]	2.333	3.000	2.667	3.000	2.667	3.000	**1.667**	3.000	3.000	3.333	2.467	3.067
rmsAAD	LMM [88]	4.000	3.667	3.667	3.000	3.667	3.000	4.000	3.333	3.333	3.000	3.733	3.200
	SVR [34]	2.667	2.333	2.667	2.333	2.667	2.667	2.667	2.667	2.000	2.000	2.533	2.400
	CB-CNN [280]	**1.000**	**1.000**	**1.000**	**1.667**	**1.000**	**1.333**	**1.667**	**1.000**	**1.667**	**1.667**	**1.267**	**1.333**
	WS-AE [11]	2.333	3.000	2.667	3.000	2.667	3.000	**1.667**	3.000	3.000	3.333	2.467	3.067

TABLE 5.6: The measured HU performance delta (quantified as Δ) in RMSE and rmsAAD for all methods. This table was also utilized in the PhD Candidate's work in [235].

Train. set size →	**66%**		**33%**		**13%**		**6%**		**1%**		**Mean**	
Metric→	**RMSE**	**rmsAAD**	**RMSE**	**rmsAAD**	**RMSE**	**rmsAAD**	**RMSE**	**rmsAAD**	**RMSE**	**rmsAAD**	**RMSE**	**rmsAAD**
LMM [88]	0.103	0.105	0.212	0.218	0.489	0.484	0.494	0.488	0.351	0.343	0.330	0.328
SVR [34]	0.944	1.004	1.542	1.672	1.722	1.887	1.572	1.687	1.181	1.239	1.392	1.498
CB-CNN [280]	1.687	1.744	3.401	3.416	2.961	2.857	2.444	2.365	1.682	1.671	2.435	2.411
WS-AE [11]	3.487	3.499	5.425	5.623	3.978	3.949	4.021	3.826	1.442	1.358	3.670	3.651

5.2.6 Concluding Remarks

This study demonstrates the critical importance of proper data stratification in HU research and highlights the superior performance of deep learning architectures when adequately trained. The CB-CNN model consistently outperformed traditional methods like LMM and other deep learning approaches across multiple benchmark datasets, particularly when leveraging spectral-spatial features that provide richer contextual information for abundance estimation. The results underscore that while classical linear mixing models may suffice for certain datasets with limited training samples, modern deep learning architectures excel when sufficient labeled data is available, emphasizing the ongoing challenge of data scarcity in remote sensing applications.

The comparison between RS and GSV reveals a fundamental methodological flaw that has likely affected numerous studies in the field. The information leakage inherent in RS approaches produces overly optimistic results that can be up to five times better than the true model performance, creating a false sense of accuracy that could lead to operational failures in real-world deployments such as satellite sensor applications. This finding calls for a paradigm shift in HU validation protocols, advocating for GSV or similar spatially-aware stratification methods to ensure realistic performance estimates. Future research should prioritize the development of data augmentation techniques and novel architectures that can effectively handle the sparse labeled data problem while maintaining rigorous validation standards that reflect true operational conditions.

5.3 Toward Compact Deep Learning for Hyperspectral Unmixing

In this work, we focus on benchmarking several literature-based architectures, a multi-feature learning approach, as well as the proposed GCNN-based models to address the task of HU. In this experimental study, we incorporate multiple multi-branch CNN models previously proposed by the PhD Candidate in [233], which integrate spectral, spatial, and spectral-spatial feature fusion to enhance the estimation of abundance vectors and are treated as a recap and reference point. Additionally, we benchmark GCNN models to evaluate the capabilities of such architectures in the context of HU. To the best of our knowledge, such topologies have not yet been explored for hyperspectral abundance vector estimation so far.

5.3.1 Multi-Branch Convolutional Neural Networks for Hyperspectral Unmixing: Reminder from the State of the Art

Our experimental results had demonstrated that the multi-branch approach outperforms other state-of-the-art classical and deep learning-based architectures in HU tasks. To complete the study, and in line with the experiments presented in Section

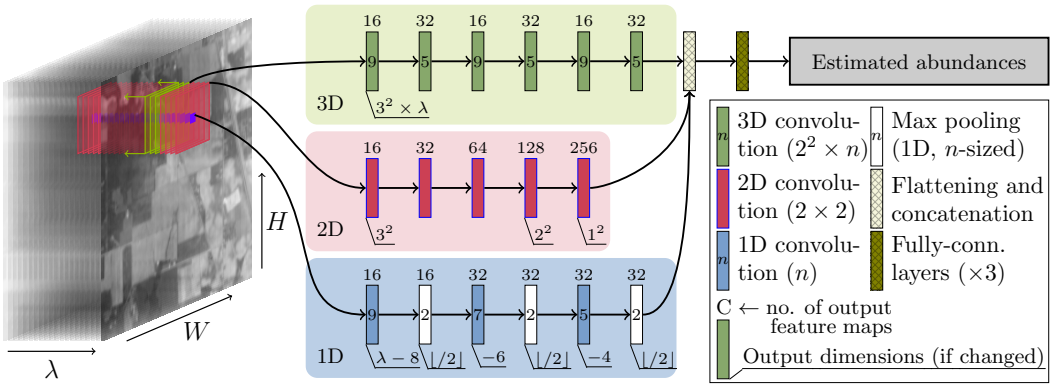


FIGURE 5.11: A high-level view of our multi-branch approach. Three branches extract spectral (1D), spatial (2D) and spectral-spatial (3D) features from an input HSI. Subsequently, those features are later fused and fed to the fully-connected regressor module, to estimate the abundances. For the 1D branch, we show a relative change in the dimensionality. This diagram was also utilized in the PhD Candidate's work in [233].

5.1 and Section 5.2, we conduct an analysis of how varying training set sizes affect the quality of the estimated abundance vectors and the generalization capabilities of the evaluated models. Finally, we assess the robustness of the larger multi-branch models by evaluating their accuracy under noise contamination.

In our proposed multi-branch approach, we utilize early and late fusion techniques coupled with a multi-branch methodology that allows the model to learn different types of fine-grained features across various domains of data. In Figure 5.11, we visualize the feature learning mechanisms of the model via this approach. The model utilizes 1D convolution to extract feature representations from the spectral dimension of the input sample of shape $p \times p \times \lambda$. The variable p indicates the spatial extent of the sliding window used to generate examples, with the central element being the pixel of interest—for this sample, the abundance vector is estimated. The variable λ denotes the number of spectral bands, which is the dimension utilized by the kernel in 1D convolution.

Consequently, the 2D convolutional layers operate by extracting features through movement across the spatial dimensions. In contrast, the 3D convolutional layers utilize both spectral and spatial dimensions simultaneously. As shown in Figure 5.11, each of the branches shares a common concatenation layer. Analogously to other models proposed in this chapter, the network concludes with several fully-connected layers that serve as a regressor to estimate the abundance vectors.

Several methods incorporating similar fusion strategies via multi-branch networks have been introduced in the literature. Audebert et al. [8] implemented such fusion in a multi-branch network. In [68], a multi-branch fusion network was utilized for hyperspectral classification. Feng et al. [64] proposed an attention-based multi-branch CNN for the same task. Cai et al. [26] introduced a multi-branch multi-scale residual fusion network, and Guo et al. [82] presented a novel approach that incorporates

weight sharing and an attention mechanism within a multi-branch network.

The underlying structure of the model consists of parallel blocks designed to extract different feature representations, which are subsequently concatenated. The first block utilizes three 1D convolutional layers, each followed by a max pooling layer to reduce the size of the activation maps. The kernel sizes in the spectral dimension are set to 9, 7, and 5 for each respective layer, with pooling filters of extent 2. This setup helps control the increasing receptive field as the depth of the network increases. The number of feature maps in each 1D convolutional layer is set to 16, 32, and 32, respectively.

As the input sample is a 3-dimensional tensor, it is reshaped to fit the 1D convolutional format by concatenating the spatial pixels into a single axis. This operation is represented as:

$$p \times p \times \lambda \rightarrow p^2 \times \lambda, \tag{5.7}$$

where $p \times p$ denotes the input patch shape. The first dimension is treated as the number of input feature maps, with the kernel sliding along the second (spectral) dimension. This operation is limited in its feature-learning capability because it only utilizes the spectral axis.

The second module, the 2D convolutional branch, extracts spatial features from the input sample. This branch uses five convolutional kernels with 2×2 spatial extent and spectral depth equal to λ. The number of feature maps is set to 16, 32, 64, 128, and 256 for the five convolutional layers, respectively.

Finally, the 3D convolutional branch includes three blocks, each containing two convolutional layers with kernel sizes of $2 \times 2 \times 9$ and $2 \times 2 \times 5$. These filters move in three dimensions to extract joint spectral–spatial features. Each block has 16 and 32 activation maps, respectively.

The model concludes with flattening and concatenation layers, followed by a three-layer fully-connected regressor with 512, 64, and c units, where c represents the number of total endmembers in the input HSI. ReLU activation is applied in all layers to mitigate vanishing and exploding gradient problems [14], [90], [192]. The output layer uses softmax activation to satisfy hyperspectral unmixing (HU) constraints: *i*) the abundance nonnegativity constraint and *ii*) the abundance sum constraint, as discussed in Section 1.2.

In our study, we further modified the baseline multi-branch CNN architecture (MB). Such modifications aim to improve the foundational topology via adjustments to the number of features, employing residual connections, and more advanced and multifold training scheme. We designed the model modifications to be model-agnostic so that they could be extrapolated into other multi-branch architectures. Below we define the added modifications in detail:

1. In the first modification variant, we tackle the phenomenon that is present in the baseline multi-branch CNN model concerning the number of output features

extracted by each parallel block. More specifically, there is a high imbalance of such activations when compared across 1-, 2-, and 3-D branches, e.g., the number of units after flattening the output of 3D module can be orders of magnitude larger compared to those from other branches. For the Urban dataset with the input shape of $3 \times 3 \times 162$, we observe approximately around 500, 200, and more than 46000 features for 1D, 2D and 3D parallel CNN components. It is highly probable that after concatenating such activations from each branch, there would be a noticeable bias in the direction of 3D branch, hence it would hinder the learning capabilities of other branches and negatively impact their effectiveness in feature extraction. Motivated by this extensive feature imbalance, we apply feature reduction in the 3D branch by employing two additional convolutional layers, without padding, and keeping the same kernel shapes (hence effectively adding one more block). Furthermore, we modify the strides of such new layers to be $1 \times 1 \times 4$ as well as $1 \times 1 \times 2$ for each layer respectively. We refer to this variant as MB-DR

2. The next modification aims to improve the gradient flow by incorporating residual connections to the model. It was experimentally proven in the literature that such operation allows the model to accelerate training process and mitigate the problems of vanishing and exploding gradient phenomena [22], [87], [96]. In this variant, we employ residual connections in the 3D branch to mitigate the mentioned problems and enhance propagation of the original hyperspectral information within the model. This modification is based on the previous advancement, i.e., MB-DR, where we incorporated additional feature reduction to the 3D branch. More specifically, the output of each consecutive block is combined via a sum aggregation with the output of the previous block, e.g., output of the second module incorporates additionally first block output, where the output from first block utilizes the repeated input data. We denote this model as MB-Res.

3. The last modification which bases on the MB-Res architecture incorporates a two-step training process: *i*) the initial training includes optimizing each branch separately by decoupling them from each other, and *ii*) fine-tuning the entire architecture or only the regressor part. Consequently, the second step has two alternatives where in the former case the training strategy incorporates a scheme where the feature extractor is pre-trained and then fine-tuned together (MB-PT). In the latter case, where only the regressor part is trained, we refer to it as MB-TL as it resembles transfer learning methodology. Furthermore, in the MB-PT, we fine-tune parameters from all branches at the same time, whereas in the MB-TL scheme, we assume the feature extractors from each branch trained separately are already properly converged and ready to be utilized, and only fine-tune the regressor part (which accounts to a smaller subset of trainable weights). Consequently, the former method is faster to execute, however has

its own limitations when utilizing the model on different dataset and can affect the generalization capabilities of the model.

Finally, in the experiments, we analyze the models in the following order, with the networks expanding their predecessors: MB→MB-DR→MB-Res→MB-PT and MB-TL.

5.3.2 Graph Convolutional Neural Networks for Hyperspectral Unmixing

The next set of models we investigate are Graph Convolutional Neural Network (GCNN)-based architectures, designed specifically for HSI abundance estimation. The primary model, referred to and depicted in figures and tables as GCNN, leverages graph-based learning techniques to effectively capture both spectral and spatial dependencies inherent in hyperspectral data. The model begins by reshaping the input hyperspectral patch of size $n \times n$, with λ spectral bands, into a sequence of n^2 nodes, where each node represents a single pixel and contains its corresponding spectral features. This formulation enables the construction of a graph in which each node corresponds to a pixel within the local spatial neighborhood, and the model performs global regression at the graph level. An alternative approach, not adopted in this work, considers the entire HSI cube as a single global graph, where each pixel (node) is assigned a regression target—typically an abundance vector. In contrast, the approach used here allows the GCNN model to process local neighborhoods individually as graph structures, thus enabling localized feature extraction and prediction.

The spatial relationships between nodes are captured by a symmetric, normalized adjacency matrix, computed based on inverse Euclidean distances between pixel coordinates within the patch, following the method of [117]. This adjacency matrix is then replicated across the batch dimension and used consistently as the graph connectivity structure throughout the model. Spectral feature processing begins with a learnable dense transformation layer followed by batch normalization and ReLU or PReLU activations, which enhances the spectral representations before graph convolutions. The model employs several graph convolutional layers, which aggregates features from neighboring nodes by multiplying node features with the normalized adjacency matrix and applying learned linear transformations. The operations consist of fully-connected layers and the weight initialization technique follows Xavier method [47]. Such initialization allows is to mitigate the problems of vanishing [90] and exploding [14], [192] gradient problems.

To further improve focus on the most informative spectral bands, a lightweight spectral attention mechanism, that is composed of two fully-connected layers is integrated into the model. It incorporates a global pooling operation across neighboring pixels. Afterwards, it employs a fully-connected layer with reduction ratio to subsample the channel space to lower dimension. Subsequently, the second dense

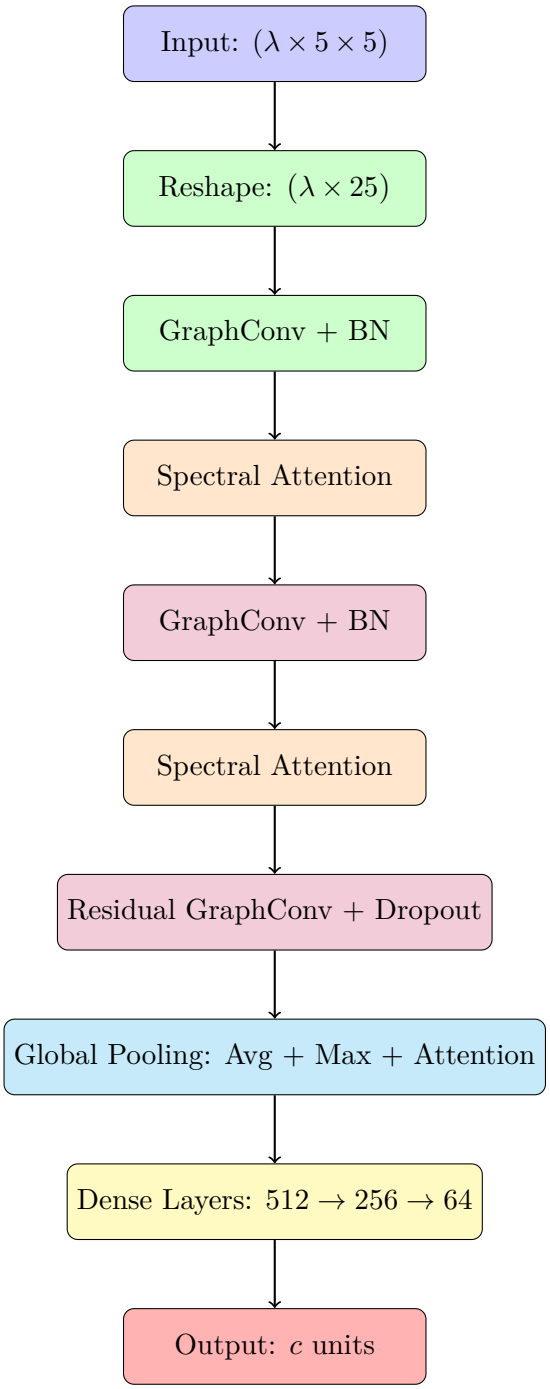


FIGURE 5.12: Simplified architecture of the proposed GCNN. The model includes spectral and graph-based modules, attention mechanisms, and multi-stage feature fusion for robust abundance estimation. Additionally, GCNN incorporates batch normalization (BN) layers to accelerate training. The example utilizes a 5×5 spatial patch, on which the adjacency matrix is created. Finally, c represents the number of endmembers in the HSI, whereas λ denotes the number of bands.

layer decodes the denser representation of spectral channels into the original projection. Such operation allows us to estimate learnable spectral-channel-wise attention weights. Finally, the output of such attention module equals to the input scaled by the computed attention weights for each spectral channel. This attention modulates

the spectral features to emphasize salient spectral information and during preliminary experiments proved to have great impact on general convergence correctness as well as training time and final abundance metrics.

The architecture extends to multi-scale spatial modeling by computing multiple adjacency matrices representing local, medium-range, and global spatial relationships, enabling multi-scale graph convolutions that capture spatial dependencies at varying distances. It is worth emphasizing that the global spatial relationships are understood as the total extent of the utilized patch neighborhood and are bounded by the established hyperparameters prior to training, thus it should be carefully selected based on the dataset. The GCNN model also incorporates batch normalization (BN) layer to stabilize and accelerate training [99]. Furthermore, the model applies residual graph convolution blocks with batch normalization and PReLU activations improve gradient flow and enable training of deeper models. Such operation allows us to diminish the possible vanishing gradient problems [90] that could arise during training deeper architectures.

At the final level of the GCNN, we apply further feature aggregation that combines global average pooling, global max pooling, and a spectral attention-weighted pooling mechanism to generate a comprehensive feature vector that captures diverse aspects of the spatial-spectral information and later is utilized in the regressor part of the model. Finally, a fully connected regression head with multiple dense layers, batch normalization, PReLU activations, and Dropout culminates in a softmax or sigmoid output layer depending on the number of classes, enabling effective multi-class or binary classification.

This refined GCNN framework effectively exploits the spatial-spectral structure of hyperspectral data, providing robust performance and flexibility across different HSI classification scenarios. Throughout the GCNN model layers, we exploit Dropout regularization technique with 0.1 and 0.2 rates, to alleviate the problem of overfitting. We refer to this model as **GCNN** and depict its general architecture in Figure 5.12 for a 5×5 spatial window.

Also, we introduce, the GCNN with Chebyshev GCN layers (referred to as **GCNN-Cheb**) is a DL architecture tailored for HSI regression and unmixing. By integrating spectral attention mechanisms with advanced spectral graph convolution, this model is specifically designed to capture intricate spatial and spectral dependencies within hyperspectral data. A distinguishing component of GCNN-Cheb is the use of **Chebyshev polynomial-based graph convolutional layers** proposed in [48]. These layers enable efficient and localized filtering on graphs by approximating spectral graph convolutions with a truncated expansion of Chebyshev polynomials of the graph Laplacian. Unlike standard GCNNs which are limited to immediate neighbors, the Chebyshev formulation allows information propagation from a K-hop neighborhood, where K is a tunable hyperparameter. This leads to more expressive and deeper spatial context modeling.

In practice, the model constructs two separate convolutional paths using Chebyshev graph convolutions (ChebConv) with different polynomial orders: $K = 3$ and $K = 5$. The first path applies a ChebConv layer with 64 output filters and $K = 3$ hops, followed by batch normalization, a PReLU non-linearity, and dropout with a rate of 0.2. Similarly, the second path employs a ChebConv layer with 64 filters and $K = 5$, also followed by batch normalization, PReLU, and dropout. This dual-path approach allows the model to capture multi-scale spatial relationships within the graph structure, aggregating information from both local and more distant neighborhoods. The outputs of both Chebyshev paths are subsequently fused with features from previous stages via an additive merge operation. This aggregated output is then propagated forward through the network. By combining multiple receptive fields and residual pathways, GCNN-Cheb retains robustness in learning hierarchical spatial features.

In addition to the graph convolutional enhancements, GCNN-Cheb also incorporates the **spectral attention module** described earlier in GCNN model. This module adaptively emphasizes informative spectral bands by applying a lightweight channel attention mechanism composed of two fully-connected layers with a channel reduction and expansion operation. This modulation of spectral features helps the network focus on the most discriminative bands, improving convergence, generalization, and overall performance. As with the base GCNN architecture, adjacency matrices are computed using normalized inverse Euclidean distances within each spatial patch and replicated across the batch. These matrices govern the connectivity for the graph convolution operations. Dropout layers with rates of 0.1 and 0.2 are consistently applied throughout the model to mitigate overfitting, particularly given the relatively small sample sizes common in HSI datasets.

A crucial aspect of the GCNN-Cheb model lies in the parameterization of the ChebConv layers. Each Chebyshev graph convolutional layer contains a learnable weight tensor of shape $(K, F_{\text{in}}, F_{\text{out}})$, where K is the polynomial order (i.e., the maximum hop distance), F_{in} is the number of input spectral features (channels), and F_{out} is the number of output filters. This weight structure allows each Chebyshev polynomial term to apply a distinct transformation to the input features, effectively encoding multi-hop neighborhood information. Consequently, the number of trainable parameters in a single ChebConv layer grows linearly with K, and quadratically with the number of feature channels. For instance, a layer with $K = 3$, $F_{\text{in}} = 128$, and $F_{\text{out}} = 64$ would contain $3 \times 128 \times 64 = 24{,}576$ parameters. Therefore, the choice of K, as well as the width of the network (i.e., number of filters), directly impacts both the model's capacity and computational cost. This parameter flexibility enables the network to scale in complexity according to the characteristics of the input data, making GCNN-Cheb highly adaptable for varying levels of spatial and spectral heterogeneity in hyperspectral datasets. We depict the architecture of the GCNN-Cheb variant in the Figure 5.13.

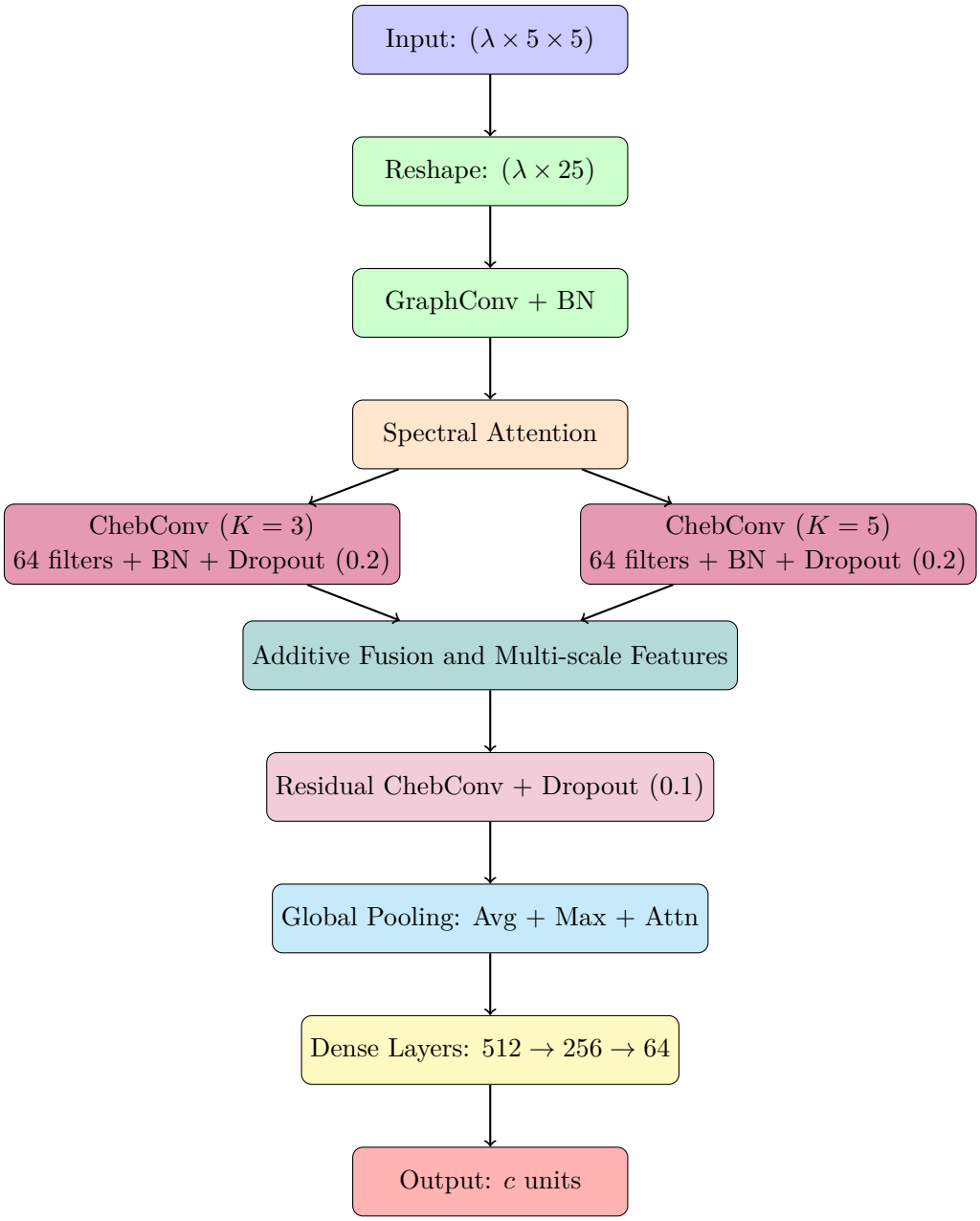


FIGURE 5.13: Architecture of the GCNN-Cheb model with Chebyshev polynomial-based graph convolutional layers. The model features dual-path Chebyshev convolutions with different polynomial orders ($K = 3$ and $K = 5$) to capture multi-scale spatial relationships, followed by additive fusion and residual processing. The Chebyshev formulation enables efficient K-hop neighborhood information propagation, providing more expressive spatial context modeling compared to standard graph convolutions. Furthermore, model incorporates batch normalization (BN) layers. Finally, c represents the number of endmembers in the HSI and λ number of spectral bands.

In summary, GCNN-Cheb advances the base GCNN framework by replacing standard GCN layers with Chebyshev polynomial GCNNs for enhanced spatial locality

and multi-hop receptive fields; introducing parallel convolutional paths with different Chebyshev orders ($K = 3$ and $K = 5$) to support multi-scale spatial feature extraction; maintaining spectral attention and residual graph convolutional blocks to ensure rich and stable feature learning; and preserving the same adjacency formulation and pooling-based feature aggregation strategy as the original GCNN. We hypothesize that this hybrid approach, combining spectral attention with multi-scale Chebyshev graph convolutions, can demonstrate significant improvements in the model's ability to regress accurate abundance maps and classify complex HSI scenes, as evidenced by the evaluation metrics reported in Subsection 5.3.6.

5.3.3 Experimental Settings

In our experiments, we pursued several key objectives. First, we investigated the impact of employing multi-branch architectures, including their various design variations, in conjunction with different training set sizes for the task of HU as a recap from our previous experiments concerning multi-branch approaches proposed by the PhD Candidate in [233]. Second, we benchmark and validate the effectiveness of the proposed multi-branch methodology by comparing it against state-of-the-art HU algorithms. Third, we examine the robustness of multi-branch CNNs under the presence of noise in hyperspectral imagery, aiming to understand their generalization capacity in realistic scenarios. Fourth, we assess the effectiveness of graph-based convolutional models, specifically GCNN and GCNN-Cheb, by quantifying their performance and evaluating their ability to capture complex spatial-spectral relationships inherent in HSI data. Finally, we analyze the sensitivity of GCNN-based architectures to variations in the size of the training dataset, providing insights into their data efficiency and stability under limited supervision.

In our experiments, we utilized Python 3.6 with Tensorflow 2.0 as the DL framework. Furthermore, the optimizer chosen to train all of the models was set to Adam [116] with learning rate of 0.001. We kept the maximum number of epochs as 100, where the early stopping condition was set to 15 epochs without improvement, conditioned on the validation loss. Moreover, the validation dataset incorporated 10% of available samples from the training data (which were consequently removed from training set to avoid redundant samples). The size of batch was set to 256 to allow the optimizer for generalized steps. In the task of HU, we utilize the mean squared error (MSE) loss, to calculate the gradient and perform back propagation. This function utilizes the ground-truth and the estimated abundance vectors generated by the benchmarked models.

For the comparative analysis with other algorithms from the literature, we selected the following HU methods:

- A LMM proposed in [88]. For the complete description of this method please refer to Section 5.2.

- An SVR with one regressor trained separately for each endmember introduced in [33]. The detailed description of this machine learning-based model was provided in Section 5.2.
- Cube-based variant of the CNN proposed in [278]. The detailed description of this architecture was given in Section 5.1.
- The WS-AE model proposed in [10]. For the detailed specification of this model please refer to Section 5.2.
- A two-step CNN-based model proposed in [201]. It incorporates a module for endmembers extraction via a geometric method with simplex volume maximization in the subspace of the utilized benchmark. Furthermore, the vectors of abundances are estimated through a deep image prior. The introduced mechanism allowed the authors to improve the model robustness to noise and outperform other state-of-the-art methods.

To allow for equal experimental conditions, each of the DNN-based architectures, i.e., all algorithms except the LMM, employs a 3D patch of shape $3 \times 3 \times \lambda$. Furthermore, we train all of the algorithms in a supervised scenario, where the abundance vectors of the training set constitute the ground-truth labels. It is important to emphasize that in spite of that fact, such models cannot estimate the kinds nor the number of endmembers in the HSI, hence this information has to be included *a priori* the experiments. In our work, we incorporate an ablation study to evaluate different variations of the multi-branch approach. Below, we provide the description of distinct types of architectures:

- MB(1D)—incorporates only the 1D convolutional branch, thus can be considered as a standard CNN model (without added branching).
- MB(2D)–utilizes sole 2D convolutional branch.
- MB(3D)—single branch incorporating 3D convolutional layers provided before.
- MB(1D+2D)—combination of 1D- and 2D-convolutional branches coupled together.
- MB(1D+3D)—1D- and 3D-convolutional branches strategy.
- MB(2D+3D)—configuration with 2D and 3D convolutional layers only.

In the experiments we performed Monte Carlo cross-validation and samples 30 pairs of training $\boldsymbol{T}$ and test $\boldsymbol{\Psi}$ sets. It is critical to emphasize that the size of test set remains constant and does not change as we change the size of the training data. Furthermore, the test set is always composed of the same examples, even with the change of $\boldsymbol{T}$ sizes. Such approach was already proposed and suggested in [278]. The magnitude of $\boldsymbol{\Psi}$ is set to 3025, 47249, and 2500 for Sa, Ur, and JR benchmarks

respectively. Moreover the sizes of $\boldsymbol{T}$ that we modify in our experiments are equal to 1, 6, 13, 33, and 66%. Such division allowed us to construct the training datasets for each variant presented in Table 5.7. The values represent the number of training pixels for each benchmark respectively.

TABLE 5.7: Sizes of the training and testing sets for each of the investigated datasets.

Size variant	Samson	Jasper Ridge	Urban
~100%	6000	7500	41000
~66%	3960	5000	31000
~33%	1980	2500	15500
~13%	780	1000	6100
~6%	360	500	2800
~1%	60	75	470
Test sizes	3025	2500	47249

5.3.4 Benchmark Datasets

In our study, we utilized the most known and established HU benchmarks. The first Samson (Sa) dataset with dimensionality $95 \times 95 \times 156$ (the last axis determines the number of bands) incorporates three endmembers: *i*) soil, *ii*) tree, and *iii*) water. The next benchmark called Urban (Ur) constitutes a HSI with shape $307 \times 307 \times 162$ as well as six endmembers: *i*) asphalt, *ii*) grass, *iii*) tree, *iv*) roof, *v*) metal, and *vi*) dirt. Lastly, we utilize the Jasper Ridge (JR) benchmark with spatial-spectral axes shape $100 \times 100 \times 198$. JR dataset incorporates four different endmembers: *i*) road, *ii*) water, *iii*) soil, and *iv*) tree.

5.3.5 Quality Metrics

To evaluate the noise robustness of the proposed architectures, we incorporate four variants of white zero-mean Gaussian noise perturbation to the test dataset, with 20, 30, 40, and 50 dB respectively. Finally, to evaluate the estimated abundance vectors against the ground-truth data, we utilize the RMSE 5.1 and rmsAAD 5.2 metrics defined in Section 5.1.

5.3.6 Experimental Results and Discussion

TABLE 5.8: The results of the two-tailed Wilcoxon tests ($p < 0.05$)—we present the number of cases (for each training set size, out of 3 HU sets) in which the confronted variants lead to obtaining statistically the same results as those by MB. We boldface the entries, in which MB obtained the statistically significantly better results for all sets. This table was also utilized in the PhD Candidate's work in [233].

Compared with ↓	**100%**	**66%**	**33%**	**13%**	**6%**	**1%**	Total
MB(1D)	**0/3**	**0/3**	**0/3**	1/3	2/3	2/3	5/18
MB(2D)	**0/3**	**0/3**	**0/3**	**0/3**	**0/3**	1/3	1/18
MB(3D)	**0/3**	**0/3**	**0/3**	1/3	1/3	3/3	5/18
MB(1D+2D)	1/3	**0/3**	**0/3**	2/3	2/3	2/3	7/18
MB(1D+3D)	3/3	1/3	1/3	2/3	3/3	3/3	13/18
MB(2D+3D)	1/3	**0/3**	**0/3**	**0/3**	2/3	2/3	5/18
Total→	5/18	1/18	1/18	6/18	10/18	13/18	36/108

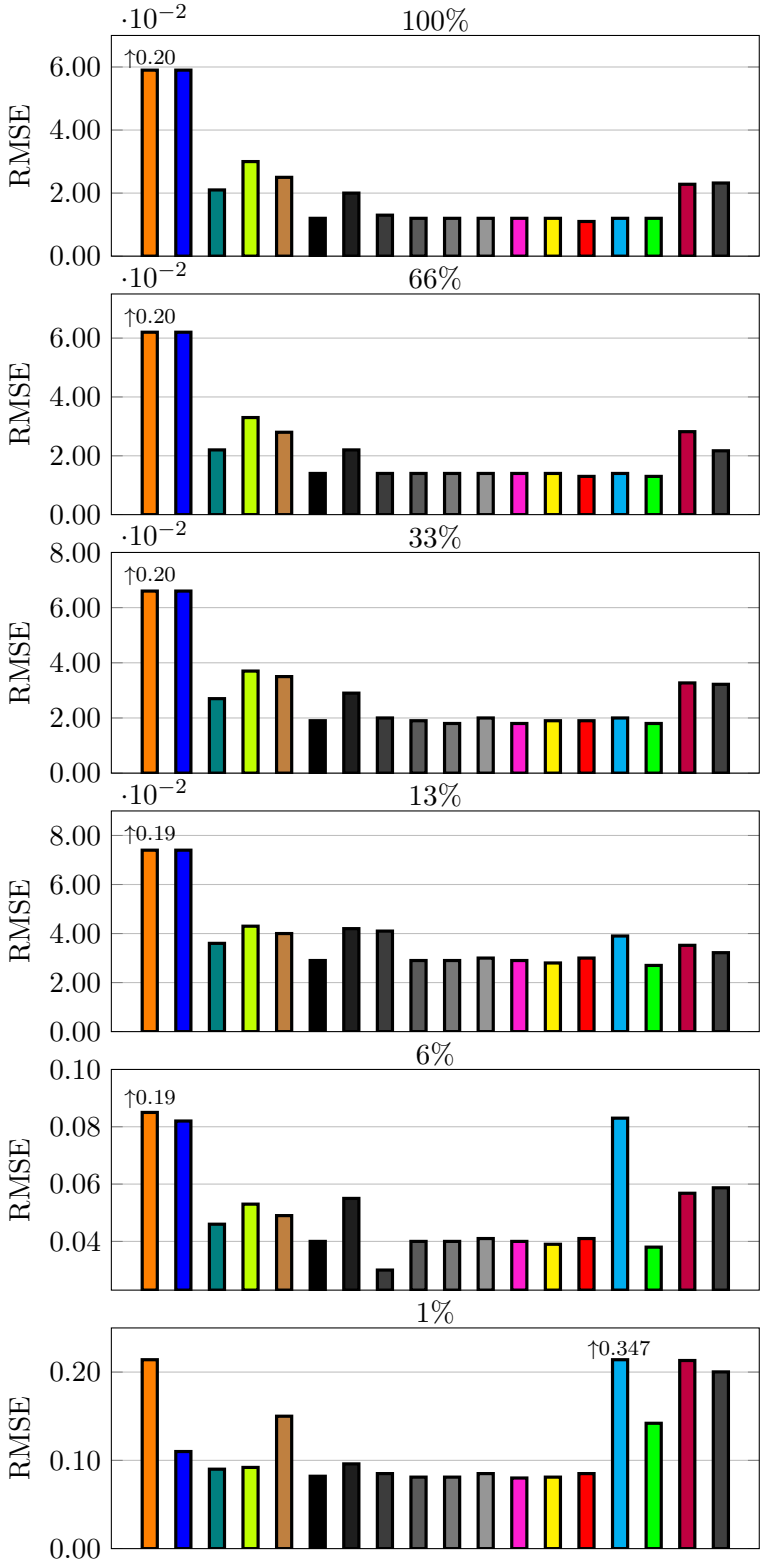


FIGURE 5.14: Overall RMSE (Ur) for different $\boldsymbol{T}$ sizes: LMM [88], SVR [34] CB-CNN [280], WS-AE [11], UnDIP [201], MB(1D), MB(2D), MB(3D), MB(1D+2D), MB(1D+3D), MB(2D+3D), MB, MB-DR, MB-Res, MB-TL, MB-PT, GCNN, GCNN-Cheb. For some methods, we indicate the exact value (out of the Y range) of RMSE above the arrow to maintain readability.

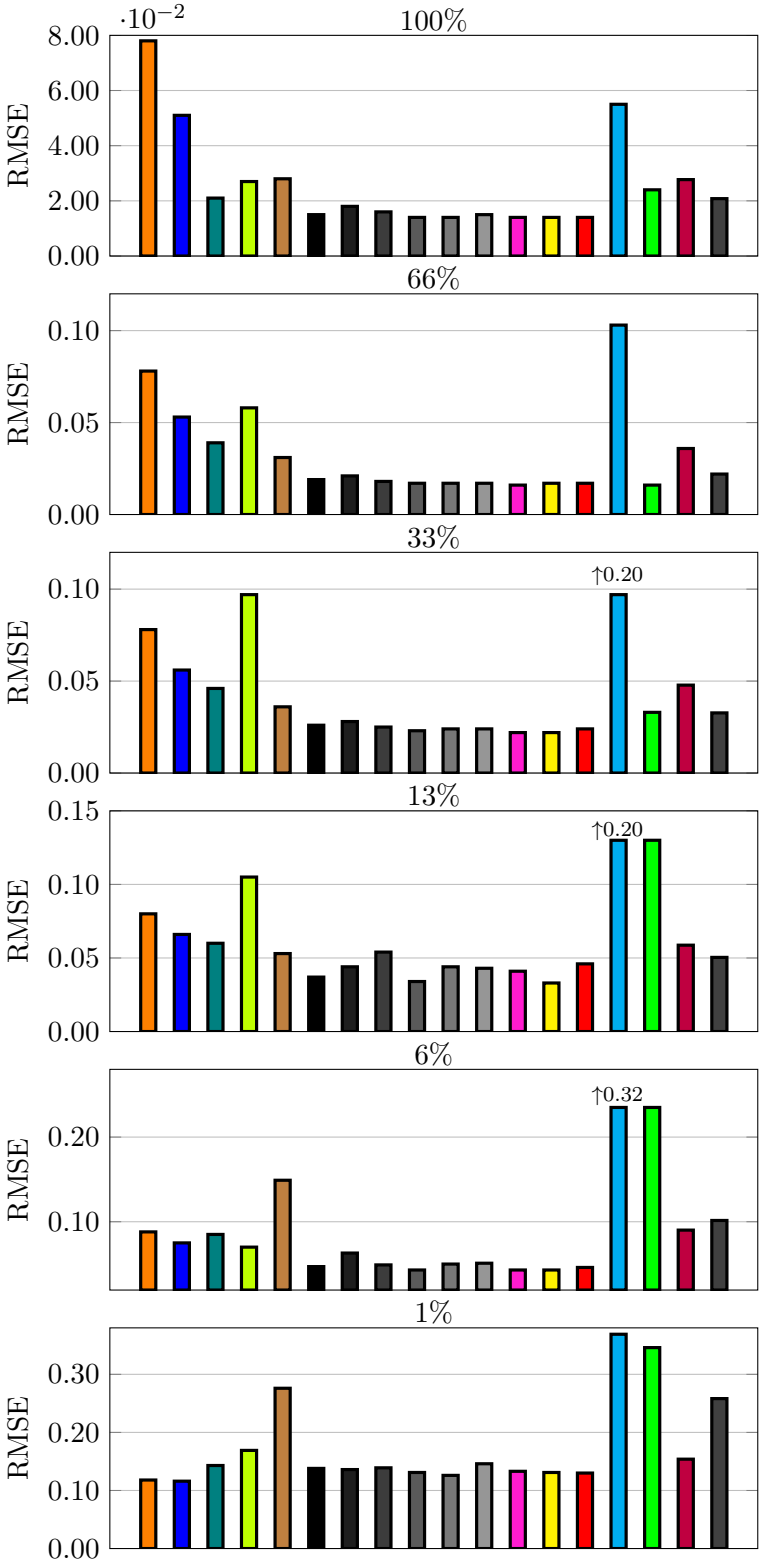


FIGURE 5.15: Overall RMSE (JR) for different $\boldsymbol{T}$ sizes: LMM [88], SVR [34] CB-CNN [280], WS-AE [11], UnDIP [201], MB(1D), MB(2D), MB(3D), MB(1D+2D), MB(1D+3D), MB(2D+3D), MB, MB-DR, MB-Res, MB-TL, MB-PT, GCNN, GCNN-Cheb.

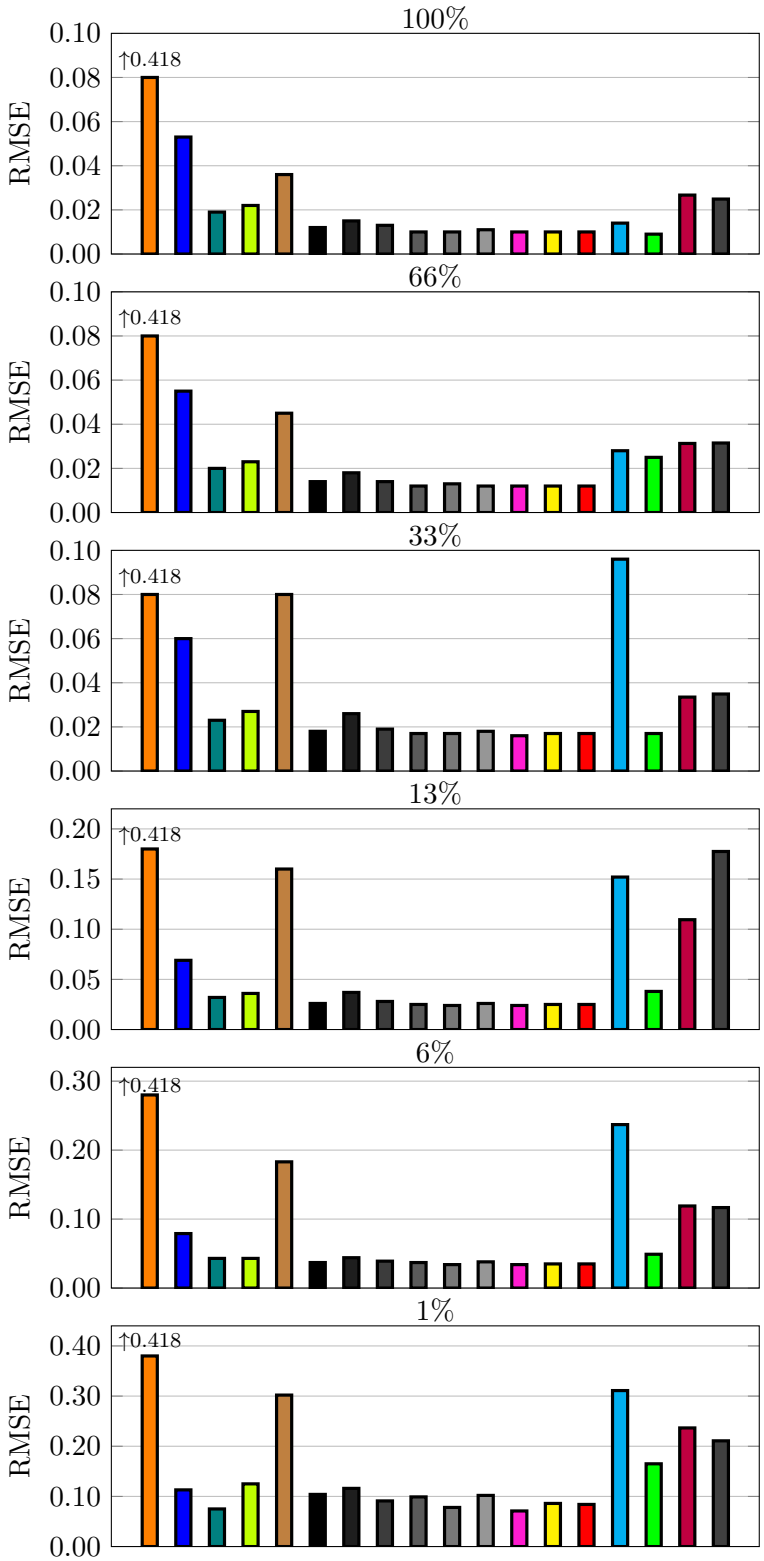


FIGURE 5.16: Overall RMSE for different $\boldsymbol{T}$ sizes for Sa dataset. Legend: LMM, SVR, CB-CNN, WS-AE, UnDIP, 1D BRANCH, 2D BRANCH, 3D BRANCH, 1D+2D BRANCH, 1D+3D BRANCH, 2D+3D BRANCH, MB, MB-DR, MB-Res, MB-TL, MB-PT, GCNN, GCNN-Cheb.

TABLE 5.9: The RMSE and rmsAAD metrics obtained for all investigated algorithms, averaged across all (30) test sets elaborated and averaged for each benchmark dataset (Sa, Ur, and JR). The best result for each $\boldsymbol{T}$ size is boldfaced, whereas the second best is underlined. The background of the globally best result (across all training set sizes) is rendered in green.

	RMSE							**rmsAAD**						
Models	**100%**	**66%**	**33%**	**13%**	**6%**	**1%**	**Mean**	**100%**	**66%**	**33%**	**13%**	**6%**	**1%**	**Mean**
LMM [88]	0.230	0.230	0.230	0.231	0.234	0.250	0.234	0.588	0.588	0.588	0.590	0.598	0.645	0.599
SVR [34]	0.054	0.057	0.061	0.070	0.079	0.113	0.072	0.132	0.138	0.151	0.175	0.199	0.292	0.181
CB-CNN [280]	0.020	0.027	0.032	0.043	0.058	0.103	0.047	0.053	0.071	0.084	0.114	0.153	0.276	0.125
WS-AE [11]	0.026	0.038	0.054	0.061	0.055	0.129	0.061	0.071	0.099	0.137	0.156	0.146	0.335	0.157
UnDIP [201]	0.030	0.035	0.050	0.084	0.127	0.243	0.095	0.075	0.085	0.121	0.191	0.304	0.615	0.232
MB(1D)	0.013	0.016	0.021	0.031	0.041	0.108	0.038	0.034	0.041	0.056	0.083	0.111	0.282	0.101
MB(2D)	0.018	0.020	0.028	0.041	0.054	0.116	0.046	0.049	0.056	0.077	0.115	0.151	0.306	0.126
MB(3D)	0.014	0.015	0.021	0.041	**0.039**	0.105	0.039	0.036	0.042	0.056	0.111	**0.103**	0.280	0.105
MB(1D+2D)	**0.012**	**0.014**	0.020	**0.029**	0.040	0.104	0.037	0.033	0.039	0.053	0.080	0.109	0.272	0.098
MB(1D+3D)	**0.012**	0.015	0.020	0.032	0.041	**0.095**	0.036	**0.031**	0.038	0.053	0.087	0.113	**0.253**	0.096
MB(2D+3D)	0.013	**0.014**	0.021	0.033	0.043	0.111	0.039	0.034	0.039	0.055	0.090	0.118	0.293	0.105
MB	**0.012**	**0.014**	**0.019**	0.031	**0.039**	**0.095**	**0.035**	**0.031**	**0.037**	**0.051**	0.084	0.106	0.255	**0.094**
MB-DR	**0.012**	**0.014**	**0.019**	**0.029**	**0.039**	0.100	**0.035**	0.032	0.039	**0.051**	**0.077**	0.105	0.263	0.095
MB-Res	**0.012**	**0.014**	0.020	0.034	0.041	0.100	0.037	**0.031**	0.038	0.053	0.090	0.111	0.263	0.098
MB-TL	0.027	0.048	0.104	0.163	0.213	0.342	0.150	0.070	0.124	0.254	0.398	0.528	0.871	0.374
MB-PT	0.015	0.018	0.023	0.065	0.107	0.217	0.074	0.038	0.045	0.059	0.164	0.270	0.545	0.187
GCNN	0.025	0.031	0.038	0.091	0.088	0.201	0.079	0.070	0.081	0.094	0.222	0.249	0.422	0.190
GCNN-Cheb	0.022	0.025	0.030	0.086	0.092	0.223	0.080	0.058	0.064	0.074	0.152	0.268	0.497	0.186

TABLE 5.10: The ranking (RMSE and rmsAAD) obtained by all algorithms, averaged across all sets (Sa, Ur, and JR). The best ranking for each $\boldsymbol{T}$ size is boldfaced, whereas the second best is underlined. For the ranking obtained for each set, see supplementary material. This table was also utilized in the PhD Candidate's work in [233].

	RMSE							**rmsAAD**						
Train. size →	**100%**	**66%**	**33%**	**13%**	**6%**	**1%**	**Mean**	**100%**	**66%**	**33%**	**13%**	**6%**	**1%**	**Mean**
LMM [88]	18.000	17.667	17.333	16.667	16.333	12.333	16.389	18.000	17.667	17.333	16.667	16.333	12.333	16.389
SVR [34]	16.667	16.333	15.667	14.333	13.333	7.667	14.000	16.667	16.333	15.667	14.333	13.000	7.667	13.944
CB-CNN [280]	11.667	12.333	11.667	10.667	10.333	7.333	10.667	11.667	12.000	12.000	10.333	10.333	7.333	10.611
WS-AE [11]	14.000	14.333	15.000	13.667	10.667	12.000	13.278	14.000	14.333	14.667	13.333	10.667	12.000	13.167
UnDIP [201]	15.333	14.000	14.333	13.000	14.333	15.333	14.389	15.000	14.333	14.333	12.667	14.333	16.000	14.444
MB(1D)	6.833	7.833	7.333	4.500	5.500	7.667	6.611	7.333	7.833	7.833	4.500	4.667	7.333	6.583
MB(2D)	10.333	10.333	11.000	10.833	11.000	10.000	10.583	10.333	11.000	11.667	11.667	11.000	10.000	10.944
MB(3D)	9.000	7.500	8.500	11.000	5.000	7.500	8.083	8.500	8.000	8.333	11.000	5.000	7.500	8.056
MB(1D+2D)	4.333	5.167	4.667	3.667	4.833	5.167	4.639	6.500	6.333	5.000	3.833	4.667	5.167	5.250
MB(1D+3D)	4.333	6.333	3.833	4.167	5.000	**2.833**	4.417	**3.167**	4.667	4.333	3.333	5.000	**3.000**	3.917
MB(2D+3D)	6.500	5.167	7.500	6.500	7.667	8.833	7.028	6.500	5.833	6.667	6.667	8.000	9.500	7.194
MB	**4.333**	**2.000**	**1.667**	3.667	**2.333**	3.000	**2.833**	2.833	**2.000**	**1.500**	4.000	**3.333**	2.833	**2.750**
MB-DR	**4.333**	5.833	4.333	**2.167**	2.667	4.667	4.000	4.500	5.000	3.333	**2.000**	**2.333**	4.667	3.639
MB-Res	**2.667**	3.500	3.333	6.167	5.333	5.333	4.389	2.667	3.000	4.000	6.333	5.000	5.000	4.333
MB-TL	11.000	11.667	14.333	15.000	17.333	17.667	14.500	11.333	12.333	15.000	15.667	17.000	17.333	14.778
MB-PT	5.000	**4.667**	5.167	10.000	10.333	14.333	8.250	5.000	**4.667**	4.333	9.333	10.667	14.333	8.056
GCNN	14.000	14.000	13.667	13.333	14.333	14.667	14.000	14.667	13.667	13.333	14.000	14.333	14.667	14.111
GCNN-Cheb	12.667	12.333	11.667	11.667	14.667	14.667	12.944	12.333	12.000	11.667	11.333	15.333	14.333	12.833

The results calculated over the benchmark datasets with varying sizes of the training sets are included in the Figure 5.14 (Ur), Figure 5.15 (JR), and Figure 5.16 (Sa), for RMSE metric indicate that utilizing all branches in our approach allows us for statistically significant improvement in the quality of delivered vectors of abundances. We provide the number of such cases, i.e., 72/108 (where this phenomenon occurred) in Table 5.8 for RMSE with p-value < 0.05 which is our confidence threshold. Furthermore, we incorporate the results obtained between all variants of the multi-branch approach, the full experimental results are provided at https://gitlab.com/jnalepa/mbhu. Our ablation study provided that when coupling the 1D and 3D branches allowed us for statistically largest improvement in HU. Such feature fusion enhances the estimated abundances vectors the most. When compared the sole 3D approach and fusion of 2D and 3D branches, there is no significant improvement of incorporating the additional 2D convolutional layers. Those two approaches lead to statistically identical HU results with RMSE as well as rmsAAD metrics in 5/18 cases. This phenomenon can be explained by the superior spectral-spatial extraction capabilities of the 3D branch. Conversely, the 2D branch fails to contribute any additional information or functionality that could improve the

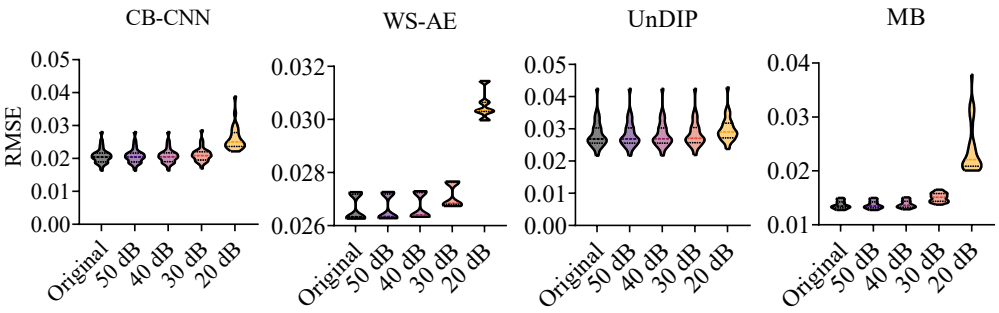


FIGURE 5.17: The impact of the white zero-mean Gaussian noise added to the original test data (Jasper Ridge) on RMSE obtained by the selected models trained from the full training set. This figure was also utilized in the PhD Candidate's work in [233].

TABLE 5.11: The impact of the patch size on RMSE (on Jasper Ridge), quantified as $\Delta_{\text{RMSE}} = \text{RMSE}^{k\times k} - \text{RMSE}^{3\times 3}$ ($k = \{5, 7, 9\}$) obtained using MB trained from the training sets of various sizes. This table was also utilized in the PhD Candidate's work in [233].

	RMSE	Δ_{RMSE}		
Train. size	3×3	5×5	7×7	9×9
100%	0.014	0.002	0.002	0.003
66%	0.016	0.002	0.002	0.003
33%	0.022	0.103	0.005	0.008
13%	0.032	0.003	0.009	0.017
6%	0.042	0.004	0.010	0.022
1%	0.105	0.028	0.055	0.038

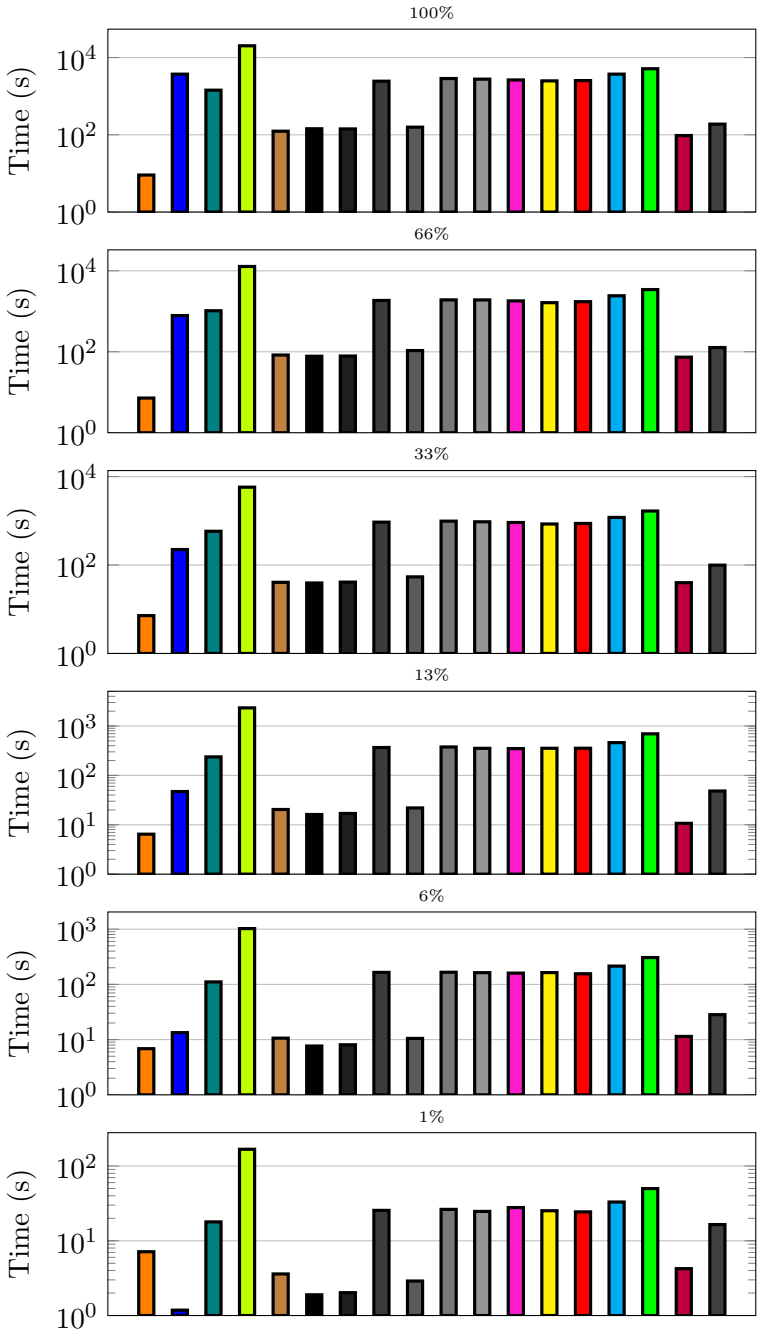


FIGURE 5.18: Average train. time for Urban (all $\boldsymbol{T}$'s sizes): LMM [88], SVR [34] CB-CNN [280], WS-AE [11], UnDIP [201], MB(1D), MB(2D), MB(3D), MB(1D+2D), MB(1D+3D), MB(2D+3D), MB, MB-DR, MB-Res, MB-TL, MB-PT, GCNN, GCNN-Cheb. This figure was also utilized in the PhD Candidate's work in [233].

unmixing quality. On the other hand, we observe significant HU improvement when combining 1D and 3D branches, hence fusing spectral and spectral-spatial features. Furthermore, the least notable enhancement in estimated abundance vectors is attributed to the smallest variant of the training set size. Such scenario identifies the

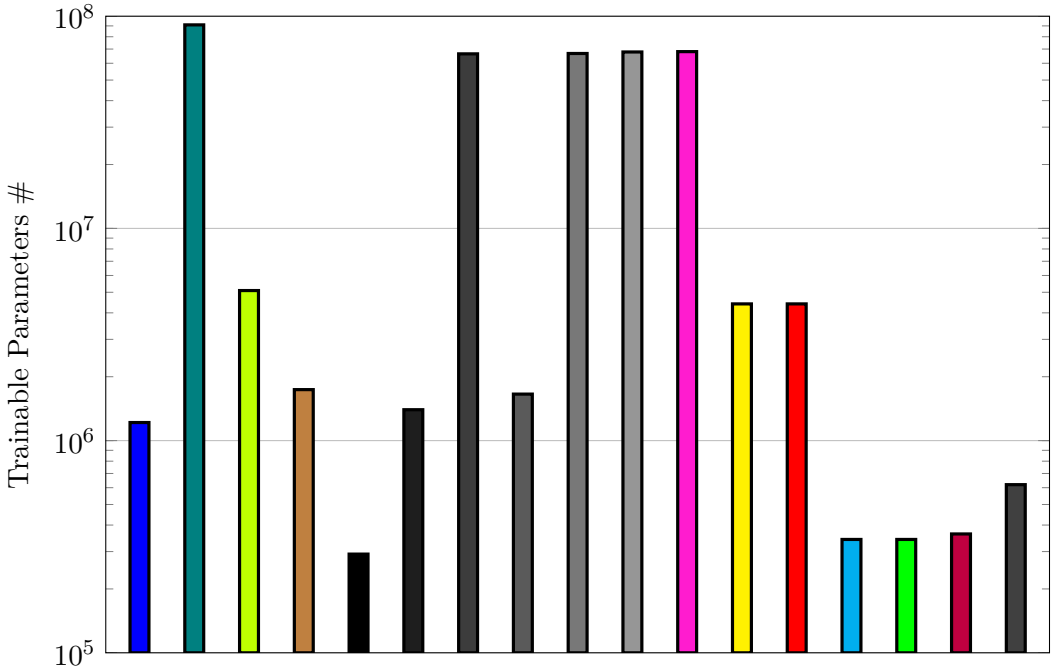


FIGURE 5.19: Number of trainable parameters for all learning-based models on the Urban dataset (log scale). LMM [88] is excluded as it is a fully constrained least-squares solution and does not involve any trainable parameters: SVR [34] CB-CNN [280], WS-AE [11], UnDIP [201], MB(1D), MB(2D), MB(3D), MB(1D+2D), MB(1D+3D), MB(2D+3D), MB, MB-DR, MB-Res, MB-TL, MB-PT, GCNN, GCNN-Cheb.

lower bound of the required amount of samples to effectively capture any dependencies between the multi-branch approaches and causes saturation effect. This issue could me mitigated by employing additional augmentation techniques via synthetic generation of data points [9].

Another interesting outcome happens when decoupling the feature extraction branches between each other in the MB-TL method. In such a scenario, the model provided the worst performance in all multi-branch cases. This can be attributed to the fact that the first part of training was not allowing the feature fusion to happen, conversely the branches could extract redundant or even interfering or disturbing features, during the concatenation layer.

In the experiments we observed that the multi-branch approaches consistently outperform other conventional and DL-based methods from the literature (Table 5.9). Furthermore, we observe such phenomenon even when aggregating the results of both metrics for all benchmark datasets. The MB and MB-DR approaches constitute the top-2-scoring methods (Table 5.10 rankings), thus proving that there is a large potential in fusing spectral and spectral-spatial features of HSI data. Such representation of features significantly impacts the capabilities of DL models when estimating the abundance vectors in HU. Moreover, we observe that for all methods the increasing size of $\boldsymbol{T}$ allows to improve the results over the test set $\boldsymbol{\Psi}$ (which remains constant

across all experiments).

Based on the experimental outcomes illustrated in Figures 5.14, 5.15, and 5.16, which correspond to the Urban, Jasper Ridge and Samson datasets respectively, a noteworthy observation emerges. Despite having a significantly lower number of trainable parameters, both the GCNN and its Chebyshev polynomial variant (GCNN-Cheb) demonstrate performance metrics that are comparable to those achieved by more complex multi-branch architectures. In Figure 5.19, we depict the total number of trainable parameters for Urban dataset for all models.

In contrast, the MB architectures typically require a substantially larger number of parameters to function effectively. Nonetheless, the simpler GCNN-based models manage to attain root mean square error RMSE scores that are similar to, and in some cases on par with, their more parameter-heavy counterparts. These findings are further substantiated by the detailed numerical results presented in Table 5.9, which highlight the comparable error levels across different model configurations.

In the results aggregated for all experiment repetitions for the multi-fold setup in Table 5.9 show that multi-branch methodology outperforms other state-of-the-art models. Furthermore, we provide the visualizations of the effectiveness in HU for all methods in the Figures 5.14, 5.15, and 5.16, which correspond to the Urban, Jasper Ridge and Samson HSIs, respectively. The visualizations are strictly and positively correlated with the experimental results utilizing the RMSE as well as the rmsAAD metrics.

In the following experiments, we evaluated the robustness of the selected models under challenging and obstructive noise conditions. Specifically, we included the multi-branch approach along with the top three best-performing models identified in the literature (see Figure 5.17). GCNN-based models were excluded from this study due to their limited model capacity and smaller architectural scale, which restricts their ability to generalize under severe noise. Their inclusion would not provide a meaningful comparison under the chosen experimental conditions. However, exploring their potential through enhanced architectures is part of our planned future work. For all multi-fold scenarios, we employ independent noise contamination of the 30 test sets $\mathbf{\Psi}$. From the experiments we deducted that UnDIP model offers the best robustness against additive Gaussian noise providing undisturbed predictions even for the largest noise configuration, i.e., equal to 20 dB. Nonetheless, the multi-branch architecture also provides steady abundance vectors almost for all noise variants, failing for the largest case of 20 dB.

In addition to the abundance errors considerations, training efficiency is another critical aspect where the GCNN models exhibit superiority. As shown in Table 5.18, both GCNN and GCNN-Cheb achieve faster training durations and quicker convergence compared to the larger and more complex MB models. This indicates a favorable trade-off between computational efficiency and predictive performance, making GCNN-based approaches attractive for scenarios with limited computational resources or time constraints.

Moreover, the architectural design of GCNNs is inherently simpler, primarily relying on fully connected layers combined with graph convolutional operations. This simplicity not only facilitates easier model maintenance and interpretability but also significantly reduces computational complexity. Such characteristics make GCNNs particularly well-suited for deployment on edge devices, such as satellites or unmanned aerial vehicles, where onboard computational resources and power consumption are severely constrained. Additionally, GCNNs lend themselves well to quantization techniques, allowing further compression of model size and acceleration of inference without significant loss of accuracy. This ease of quantization and deployment enables real-time processing and adaptability, which are critical for remote sensing applications requiring timely and reliable hyperspectral image analysis.

While the unmixing quality of GCNN-based models does not quite match the higher precision typically achieved by more complex multi-branch architectures, GCNNs offer significant advantages in terms of simplicity and interpretability. Their straightforward structure facilitates explainable AI approaches, making it easier to understand and analyze model decisions and feature importance. This transparency is particularly valuable in hyperspectral imaging applications where interpretability can be as critical as predictive accuracy.

In Table 5.11, we provide the experimental results when comparing different spatial sizes of the input patch for RMSE metric obtained for the Jasper Ridge dataset over the five test sets. In all scenarios the underlying model was selected as the MB architecture. The results indicate that increasing the spatial neighborhood window causes the model to provide less stable results, thus increasing the standard deviation to 0.220, 0.209, and 0.215 for 5×5, 7×7, and 9×9 spatial shapes respectively. Conversely, the patch of 3×3 allowed us to decrease the standard deviation to 0.002. Such phenomenon can be attributed to large ground sampling distances between the hyperspectral sensor and the earth, thus increasing the amount of noise in the input patch and incorporating more materials.

In Figure 5.18, we provide the average training time for the Urban dataset (which is highly consistent and correlated to other benchmarks). It is visible that the multi-branch architectures situate at the upper range of that metric, nonetheless, the WS-AE model constitutes the longest training time. As could be expected the LMM provides the fastest training time. It is important to emphasize that the multi-branch architecture offer similar training times as other DL-based models with clear advantage in the estimated abundance vectors in HU.

5.3.7 Concluding Remarks

In this study, we conducted a comprehensive evaluation of multiple deep learning models, including conventional, graph-based, and multi-branch architectures, for HU. Our experimental framework encompassed a wide range of benchmarking scenarios, incorporating various dataset sizes, noise levels, and architectural configurations.

The multi-branch architecture, developed in previous work by the PhD Candidate in [233], continues to demonstrate superior performance in abundance estimation. By fusing spectral and spectral-spatial features extracted through 1D, 2D, and 3D convolutional branches, the model effectively captures complex material interactions. This fusion yields statistically significant improvements in the RMSE and rmsAAD metrics across multiple datasets. Nonetheless, the performance gain saturates for the smallest training sets, suggesting a need for sample augmentation. Furthermore, an ablation study revealed that the combination of 1D and 3D branches yields the most significant improvements, while the addition of the 2D branch did not provide statistically distinguishable benefits, likely due to redundancy and insufficient spatial discrimination.

On the other hand, one of the most notable findings from our study pertains to GCNNs and their Chebyshev variant (GCNN-Cheb). Despite having a significantly reduced number of trainable parameters and a simpler architectural design, these models achieve HU performance comparable to that of SOTA and computationally intensive architectures. In addition to their effectiveness in unmixing, GCNNs exhibit marked advantages in terms of training efficiency. As shown in our runtime analysis (Figure 5.18, GCNN-based models converge more rapidly and require substantially less training time compared to deep, multi-branch counterparts. This efficiency, combined with their lower complexity (Figure 5.19) positions GCNNs as a particularly attractive option for deployment on resource-constrained platforms such as satellites. Moreover, the interpretability of GCNN architectures offers unique advantages for explainable AI (XAI) approaches in remote sensing.

In conclusion, while multi-branch architectures remain powerful tools for high-precision HU, GCNN-based models present a strong and efficient alternative. Their low computational overhead, robustness to noise, rapid convergence, and deployment versatility make them highly suitable for practical applications in operational and edge environments. Future work will explore architectural extensions to GCNNs to enhance their expressive capacity and further close the gap with complex SOTA models.

5.4 CANNIBAL: Band Selection for Hyperspectral Data

In this section, we focus on the topic of feature (band) selection in the context of the task of HU. The HSI images are constructed via capturing a continuous range of electromagnetic spectrum. Subsequently, depending on the spectral resolution of the utilized sensor—the better the sensor, the narrower the wavelengths of consecutive bands are—hundreds of channels are acquired [213]. It is important to emphasize that the qualification of sensor as *hyperspectral* refers to the continuousness and narrowness of the captured bands rather then the resulting number of channels of the underlying sensor.

Given the fact that the continuous hyperspectral bands are locally correlated between each other, (as was experimentally proven in multiple works [62], [84], [150], [161], [193], [249]) we hypothesize that by discovering the dependencies and strengths of dependencies between them, would allow us for efficient and accurate feature selection, thus lowering the resulting number of channels in the HSIs. Furthermore, we hypothesize that such selection would incorporate a mixture of representative and informative bands, without any statistically significant loss in the quality of the estimated abundance vectors in HU task.

Finally, as this dissertation focuses specifically on the hyperspectral unmixing task, we present and analyze results exclusively within this domain. It should be noted that the CANNIBAL algorithm has also been investigated for hyperspectral image segmentation tasks, as detailed in the PhD Candidate's work [240]. However, for the purpose of maintaining focus and ensuring appropriate scope within this thesis, the segmentation results are omitted from the current discussion. This targeted approach allows for a more thorough and comprehensive analysis of the algorithm's performance in the context of abundance estimation, which remains the primary objective of this research.

5.4.1 Proposed Approach for HSI Feature (Band) Selection

To tackle the tasks of HU and band selection, we employ a genetic algorithm with the linkage learning approach (GAwLL) [229] that, as a side product of the optimization, produces a weighted variable interaction graph (wVIG). Unlike in the vast majority of genetic algorithm (GA)-based methods, in our approach, we do not utilize the main product of GAwLL, instead we incorporate the side product, which later undergoes unsupervised clustering, to determine the most informative set of bands. In the domain of GAs, we are not aware of any study that discards the main product of the optimization and utilizes the side product instead. Furthermore, we employ the optimization to support the linkage learning process. Naturally, as GA progresses in generations, the individuals in the underlying populations volume up to be more representative, providing better fitness values and hence becoming of better-quality as the solution to the optimization problem. As the objective function, we selected the MSE metric 5.8, which can be formulated as:

$$\text{MSE} = \frac{\sum_{i=1}^{N}(\boldsymbol{a}_i - \hat{\boldsymbol{a}}_i)^2}{N}, \tag{5.8}$$

where the $\boldsymbol{a}$ and $\hat{\boldsymbol{a}}$ denote the ground-truth as well as the estimated abundance vectors. Furthermore, variable N refers to the size of the test set. It compares and measures the distance between the ground-truth and estimated abundance vectors. This function treats all of the endmembers equally, without any bias. In the Figure 2.2, we visualize our approach and describe each step in the method.

In our main contribution in this work, we incorporate the use case of a wVIG that later undergoes unsupervised clustering, followed by a band selection algorithm.

That algorithm performs a *greedy* feature selection based on the strengths of the interaction between the variables. The architecture of our algorithm for band selection is highly modular and customizable, e.g., the unsupervised clustering step and offers extensive flexibility. Furthermore, the number of CANNIBAL output bands is parametrizable but also can utilize non-parametric clustering. In such scenario, the number of selected features is automatically adjusted, without any user interaction. The ability of parametrization of the output number of bands may be highly desired in cases when it is crucial to perform aggressive data reduction, e.g., various satellite sensors to increase the speed of data downlink. Moreover, the proposed CANNIBAL method is task-agnostic and could be utilized in HU as well as segmentation. Finally, we provide the full software and the entire implementation at https://github.com/V-o-y-a-g-e-r/CANNIBAL/.

Linkage learning in GAs is a process of identifying and preserving real and existing beneficial combinations between genes, i.e., variables. Such phenomenon is crucial in scenarios where the fitness of a solution is not determined by the individual sole genes but groups of variables that interaction with each other (possibly in non-linear cases). The term *linkage* refers to the degree of relation or interaction between different variables in the chromosome. High linkage determines that certain variables work better when incorporated together. The linkage learning concept is utilized in different optimization problems, i.e., binary [196], [228], non-binary [195], and continuous [120] optimization. There are two main methods for learning and extracting the linkage between variables: *i*) statistical linkage learning, and *ii*) empirical linkage learning. In the former approach, the dependencies are detected via the entropy, however, it may detect false linkage. Such false positives may negatively impact the quality of the solution. To alleviate that, the empirical linkage learning approach focuses on exploiting only real dependencies. Moreover, it was proven that such methods cannot detect false linkage. In the literature, several approaches also target non-linear dependencies and incorporate monotonicity checks. The final individual is based on the fact that the near-local-optima solutions are more likely to render a positive non-monotonicity check. In the work [229], the technique of wVIG was introduced. wVIG couples the advantages of empirical linkage learning with the idea of the strength of dependency derived from statistical linkage learning, whilst not detecting false positive relationships between variables. Moreover, wVIG may be incorporated to any optimization domain, including segmentation or unmixing tasks—we utilize this advantage in our approach.

As depicted in the Figure 2.2, CANNIBAL allows to obtain three different machine learning models:

- $\boldsymbol{M}_{\text{GAwLL}}$ constitutes the main output of GAwLL. It incorporates the set of variables, i.e., bands that were selected utilizing the best candidate solution. It is important to emphasize that the final number of features here cannot be parametrized and hence is not controllable by the user.

- $\boldsymbol{M}'_{\text{GAwLL}}$ refers to any model that utilizes the bands selected by GAwLL but is trained from scratch, after obtaining the candidate solution $\boldsymbol{M}_{\text{GAwLL}}$.
- $\boldsymbol{M}_{\text{CANNIBAL}}$ constitutes any model (incorporating segmentation or unmixing) trained over the selected bands by CANNIBAL. It is important to emphasize that the resulting number of features here is controllable by the user. The selection can be parametrizable or run without parameters, when incorporating non-parametric clustering algorithms, e.g., Affinity Propagation [55].

In this work, the wVIG structure is utilized to store a weighted graph with its weights referring to the strength of interactions among input variables. In the case of this research, the input variables correspond to the hyperspectral bands but can be easily extrapolated to any other task, requiring feature selection. Given an optimization task where the fitness function can be denoted as $f(\mathbf{x})$, with $\mathbf{x}$ being the vector of a single candidate solution, the interaction between two independent, i.e., decision variables can be formulated as scenario where variables x_g and x_h interact, if the impact on $f(\mathbf{x})$, of varying x_h is contingent on of x_g. As previously mentioned, GAwLL extracts an empirical wVIG, which is a side-effect of the optimization performed by a GA. Therefore, constituting it a no-cost empirical linkage-learning technique (as it is conducted during the optimization time). Based on a candidate solution $\mathbf{x} \in \mathbb{B}^N$, the differences in fitness can be formulated as:

$$\delta_g(\mathbf{x}) = f(\mathbf{x} \oplus \mathbf{1}_g) - f(\mathbf{x}) \tag{5.9}$$

and

$$\delta_g(\mathbf{x} \oplus \mathbf{1}_h) = f\Big(\mathbf{x} \oplus \mathbf{1}_h \oplus \mathbf{1}_g\Big) - f(\mathbf{x} \oplus \mathbf{1}_h), \tag{5.10}$$

where $\oplus$ refers to the bitwise XOR function, $\mathbf{1}_g \in \mathbb{B}^N$ is a solution characteristic vector with the *g-th* element equal to one whilst all other elements are equal to zero. Furthermore, $f(\mathbf{x} \oplus \mathbf{1}_g)$ refers to the fitness of candidate solution $\mathbf{x}$ after flipping x_g variable, and $f\Big(\mathbf{x} \oplus \mathbf{1}_h \oplus \mathbf{1}_g\Big)$ denotes the fitness of a candidate solution $\mathbf{x}$ after flipping both variables x_h and x_g. Based on the fitness differences, we can define:

$$\omega_{g,h}(\mathbf{x}) = |\delta_g(\mathbf{x} \oplus \mathbf{1}_h) - \delta_g(\mathbf{x})|. \tag{5.11}$$

Following the theorem 3.1 in [229] it indicates that if $\omega_{g,h}(\mathbf{x}) > 0$, then there is an interaction between variables x_g and x_h, which cannot be a false positive dependency.

The offspring population in GAwLL us generated solely by mutation operation, where three children are created. The first and second offspring are equal to the parent individual with the exception of inverted bits in the chromosome, i.e., g and h respectively. The indices are randomly sampled from a uniform distribution and only one change per new individual is allowed. The third child is generated via flipping both selected genes, i.e., g and h. This scenario allows us to obtain all possible combinations considering bits of g and h, and are later utilized to compute

the strength of interaction $\omega_{g,h}$, between such variables. The interaction can also be aggregated over a set of new offspring generated by mutation. Finally, the wVIG is iteratively built, during the passing generations of GAwLL, with the initial graph incorporating no edges, and each vertex equal to one variable, i.e., hyperspectral band.

In the case of HSIs, the candidate vector incorporates the solution of $\mathbf{x} \in \mathbb{B}^N$, where N indicates the number of channels. The GAwLL model incorporates a KNN model, with k parameter set to three and the fitness function is denoted as:

$$f(\mathbf{x}) = 0.98 f_1(\mathbf{x}) + 0.02 \frac{N - \sum_{i=0}^{N-1} x_i}{N}, \tag{5.12}$$

where $f_1(x)$ is a metric measuring the performance of the base machine learning model in a particular task, whereas the second component could be interpreted as the selection rate and serves the role of a regularizer. The fitness evaluation is deterministic based on the combination of variables in the candidate solution. As the metric for measuring quality of the solution, the rate of correctly classified pixels is used for HSI segmentation, whereas the MSE function is employed in HU. It is important to emphasize that the wVIGs are estimated per target (in multi-target tasks, there can be more than one wVIG). Consequently, to obtain a wVIG that is agnostic to the number of targets, e.g., endmembers in HU task, we sum all of the obtained wVIGs and obtain a holistic view on inter-variable dependencies across all of the targets.

The time complexity of CANNIBAL (excluding the clustering algorithm) amounts to $\mathcal{O}(\sum_{i=1}^{k} s_i^2)$, where s_i denotes the number of bands in the cluster i and k refer to the total number of clusters.

The subsequent operation is the unsupervised clustering of the aggregated wVIG. Such grouping allows us to identify strong interactions between independent variables. Our core hypothesis is that selecting a single representative band from a cluster of strongly interacting bands will preserve the overall information content of the cluster. Consequently, the wVIG structure could be interpreted as an adjacency matrix, which designate the strengths of relations between its members. More specifically, we aim to find a division of the wVIG, where the variables between different clusters have low strengths of interactions. On the other hand, we aim to construct such groups of which bands relate strongly to each other. The clustering process visualization is depicted in the Figure 5.20. After the clustering operation, CANNIBAL selects the representative of each group. To select the best representative, we calculate the total similarity between other variables (in the same cluster) for each band. Subsequently, we pick the band with the highest total similarity as the final cluster representative. This process allows us to capture the essential representation and preserve as much as possible information from each cluster, by selecting only one band. Furthermore, this approach utilizes empirically-learned linkages, hence assures that the selected variables are indeed the best representatives in terms of total similarity for each

cluster. We refer to this step as *cannibalization*, since only a single band of each cluster prevails.

5.4.2 Experimental Settings

In our experiments, we focus on validating the impact of feature selection offered by CANNIBAL with different reduction values. Furthermore, we evaluate the quality of the machine learning models that exploit bands extracted via CANNIBAL and other state-of-the-art band selection methods. Additionally, we investigate and measure the flexibility of our method in different downstream tasks, i.e., hyperspectral segmentation and unmixing. As the baseline band selection methods we incorporate in the study the following methods:

- A multi objective artificial immune system-based algorithm [274]. The best candidate solution is selected based on two objective functions, where the first one measures the quality of the solution, whereas the second estimates the amount of redundancy in the incorporated independent variables, i.e., hyperspectral bands.
- Approach that based on information theory and entropy of each hyperspectral band [81]. It incorporates the estimation of mutual information between each band and the ground-truth data. The mutual information requires to calculate the joint and individual entropies of pairs of random variables.
- A baseline method where the subset of features, i.e, bands is randomly sampled with a uniform distribution, to avoid any bias for the band selection.

The parametrization of each method is specified in the corresponding research. To ensure reproducibility of experimental findings as well as conclusions and robustify the results, we repeat each of the experiments 30 times. Each experiment evaluation utilizes a random forest model with different seed. For all experiment runs, we set the number of estimators in the random forest model to 100 and keep it constant. We sample the $\boldsymbol{T}$ and Ψ randomly, utilizing a pixel-based approach. Following this strategy, we ensure that there is no information leak across both datasets. Furthermore, to incorporate high representativeness of the endmembers target distributions, we perform stratified random sampling to establish training and test sets. In the case of unsupervised clustering algorithm, we employ the spectral clustering method.

5.4.3 Benchmark Datasets

In the HU experiments, we utilize the Urban dataset which incorporates 307×307 pixels and 162 hyperspectral bands. It attributes a 2 m sampling distance with a spectral resolution of 10 nm. The range of wavelengths starts with 400 nm and continues up to 2500 nm. The Urban benchmark incorporates six various endmembers: *i*) asphalt, *ii*) grass, *iii*) tree, *iv*) roof, *v*) metal, and *vi*) dirt. The training sets

contain 6000 random pixels, 1000 per endmember. We select the abundances to be between 0.2 and 0.8 to eliminate outlier cases from the training set. Consequently, the remaining $307^2 - 6000$ samples constitute the test set.

5.4.4 Quality Metrics

To measure the quality of estimated abundance vectors, we utilize the mean absolute error (MAE), MSE, explained variance (EV), as well as the coefficient of determination (R^2). The MAE and MSE metrics should be minimized, where zero indicates the perfect HU solution. Furthermore, the EV and R^2 measures should be maximized, with one being the perfect unmixing score. Additionally, we incorporate the mean (M), coupled with the minimum and maximum (MinMax) values estimated for 30 random forest models, trained with different seeds. It is important to note that there is not seed for KNN since the implementation of this model was deterministic.

5.4.5 Experimental Results and Discussion

In the Figure 5.21, the quality of selected features of $\mathcal{B}_R = 5, 10, \ldots, \mathcal{B}$ bands, measured as the HU scores utilizing MAE, MSE, EV and R^2 metrics are visualized. These metrics can be formulated as following:

$$\text{MAE} = \frac{\sum_{i=1}^{|\Psi|} |p_i - \hat{p}_i|}{|\Psi|}, \tag{5.13}$$

$$\text{MSE} = \frac{\sum_{i=1}^{|\Psi|} (p_i - \hat{p}_i)^2}{|\Psi|}, \tag{5.14}$$

$$\text{EV} = 1 - \frac{\text{Var}\,(p - \hat{p})}{\text{Var}\,(p)}, \tag{5.15}$$

$$\text{R}^2 = \frac{\sum_{i=1}^{|\Psi|} (p_i - \hat{p}_i)^2}{\sum_{i=1}^{|\Psi|} (p_i - \bar{p}_i)^2}. \tag{5.16}$$

Our proposed approach (CANNIBAL) outperforms all state-of-the-art feature selection methods in terms of EV and R^2 metrics for almost all $\mathcal{B}_R$ sets, whilst incorporating the KNN model. Furthermore, the MAE and MSE metrics showcase that CANNIBAL performs worse than BOMBS approach and random sampling strategy for the lowest numbers of selected bands, i.e., 5 up to 20. However the mentioned differences in MAE and MSE metrics are very low, when compared with the ones obtained for the coefficient of determination visible in the Figure 5.21. Such subtle differences in MAE and MSE may be attributed to the outlier cases of abundances incorporated within the Urban benchmark.

In our work, we emphasize that CANNIBAL can effortlessly be executed as non-parametric method. Furthermore, the output number of selected bands can be automatically determined utilizing non-parametric unsupervised clustering methods.

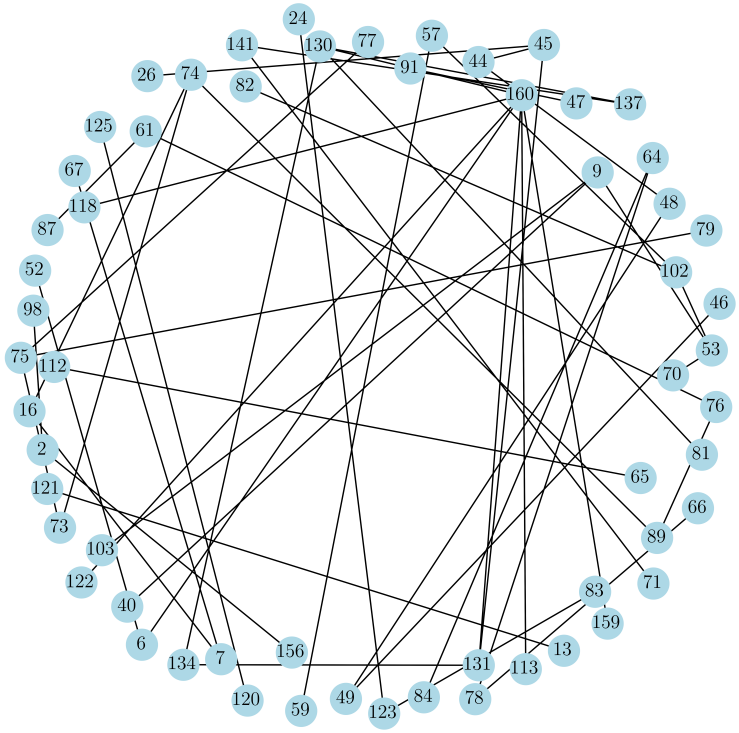
This extremely useful feature, offering flexibility is denoted in the Figure 5.21 where we incorporate the results when employing the Affinity Propagation (AP) clustering method [66] using the AP subscript. When incorporating the AP method, the dimensionality reduction rate is equal to 6.5, conversely obtaining 25 bands. As can be seen, the metrics obtained for this configuration indicate high-quality HU, without any significant deterioration of the scores. Furthermore, in the Figure 5.23, we include the normalized HU errors, calculated over the best random forest models for two endmembers, trained by utilizing the subsets of selected bands. Notably, CANNIBAL allows us to obtain a significantly lower number of features equal to 35. Moreover, there is lack of increased HU errors, when compared to other state-of-the-art methods.

Furthermore, in Table 5.13, we conducted pairwise Wilcoxon statistical tests over the aggregated quality metrics across all of the 30 random forest seed configurations, assess and measure the statistical significance of the HU scores estimated from different subsets of hyperspectral bands. The test's results indicate that the HU performance for configuration greater than 25 bands selected by CANNIBAL is statistically equivalent to that of the full or larger spectrum. Such observation proves that the bands that are excluded from clusters, and hence undergo "cannibalization" process, incorporate redundant information. From the statistical tests, we also conclude that our method consistently outperforms other feature selection methods with statistical significance.

TABLE 5.12: The results of the Wilcoxon tests verifying if the differences between CANNIBAL and other investigated techniques are statistically significant at $p < 0.05$. The background of statistically significant differences is rendered in green, and the corresponding p-values are boldfaced. This table was also utilized in the PhD Candidate's work in supplement of [240].

	MAE			MSE			R^2		
$\mathcal{B}$	BOMBS	MI	Random	BOMBS	MI	Random	BOMBS	MI	Random
5	**0.009**	**0.000**	**0.027**	**0.000**	**0.000**	**0.027**	**0.000**	**0.000**	**0.000**
10	**0.000**	**0.000**	0.053	**0.000**	**0.000**	0.123	**0.000**	**0.000**	0.154
15	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.036**
20	**0.000**	**0.000**	**0.000**	**0.000**	0.812	**0.000**	**0.001**	**0.000**	**0.002**
25	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
30	**0.000**	**0.000**	0.546	**0.000**	**0.000**	0.452	0.114	**0.000**	**0.000**
35	0.596	**0.000**	**0.000**	0.522	**0.000**	**0.000**	**0.011**	**0.000**	**0.000**
40	0.571	**0.000**	**0.002**	0.898	**0.000**	**0.002**	**0.033**	**0.000**	**0.000**
45	**0.000**	**0.000**	0.648	**0.000**	**0.000**	0.388	**0.000**	**0.000**	0.123
50	**0.001**	**0.000**	**0.006**	**0.000**	**0.000**	**0.003**	0.114	**0.000**	**0.000**
55	0.330	**0.000**	**0.000**	0.154	**0.000**	**0.000**	0.475	**0.000**	**0.000**
60	0.927	**0.000**	**0.005**	0.927	**0.000**	**0.001**	**0.008**	**0.000**	**0.000**
65	0.245	**0.000**	**0.000**	0.368	**0.000**	**0.000**	**0.001**	**0.000**	**0.000**
70	0.701	**0.000**	**0.002**	0.784	**0.000**	**0.002**	**0.019**	**0.000**	**0.000**
75	0.368	**0.000**	**0.000**	0.202	**0.000**	**0.000**	**0.011**	**0.000**	**0.000**
80	0.294	**0.000**	**0.000**	0.349	**0.000**	**0.000**	0.154	**0.000**	**0.000**
85	**0.007**	**0.000**	**0.003**	**0.048**	**0.000**	**0.002**	0.596	**0.000**	**0.000**
90	**0.001**	**0.000**	**0.036**	**0.002**	**0.000**	**0.007**	**0.006**	**0.000**	**0.000**
95	**0.003**	**0.000**	0.053	**0.030**	**0.000**	**0.024**	0.154	**0.000**	**0.000**
100	**0.036**	**0.000**	**0.011**	**0.015**	**0.000**	0.053	0.053	**0.000**	**0.002**
105	**0.033**	**0.000**	**0.006**	**0.048**	**0.000**	**0.004**	**0.019**	**0.000**	**0.000**
110	**0.012**	**0.000**	**0.003**	**0.003**	**0.004**	**0.003**	**0.009**	**0.000**	**0.000**
115	0.202	**0.004**	**0.002**	0.177	**0.009**	**0.001**	0.133	**0.000**	**0.000**
120	0.261	**0.048**	**0.001**	0.216	0.133	**0.002**	**0.021**	**0.000**	**0.000**
125	**0.004**	0.498	**0.004**	**0.017**	0.294	**0.004**	**0.021**	**0.033**	**0.000**
130	0.701	0.312	**0.040**	0.869	0.349	**0.036**	0.430	0.105	**0.001**
135	0.202	**0.001**	**0.007**	0.076	**0.004**	**0.002**	0.245	**0.001**	**0.001**
140	0.245	**0.012**	**0.000**	0.189	**0.040**	**0.000**	0.294	**0.000**	**0.000**
145	0.522	**0.001**	**0.015**	0.812	**0.004**	**0.011**	0.812	**0.001**	**0.000**
150	0.869	**0.019**	**0.006**	0.784	**0.009**	**0.006**	0.674	**0.002**	**0.001**
155	0.927	**0.012**	**0.000**	0.571	**0.003**	**0.000**	0.368	**0.027**	**0.000**
160	0.956	0.177	**0.000**	0.784	0.133	**0.000**	**0.033**	0.058	**0.000**

(a) Example weighted Variable Interaction Graph



(b) Example clustering of the Variable Interaction Graph

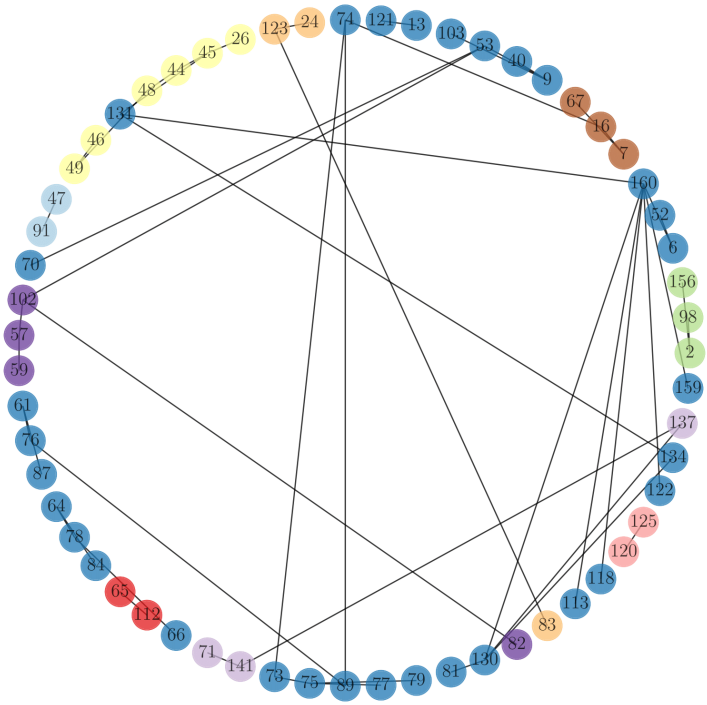


FIGURE 5.20: The (a) example of wVIG obtained by CANNIBAL for the Urban dataset—the vertices to the hyperspectral bands, whereas the edges to the uncovered interactions between the variables. In (b) we depict the clustering of VIGw. Each color represents a different cluster of bands. This figure was utilized in the PhD Candidate's work in [240].

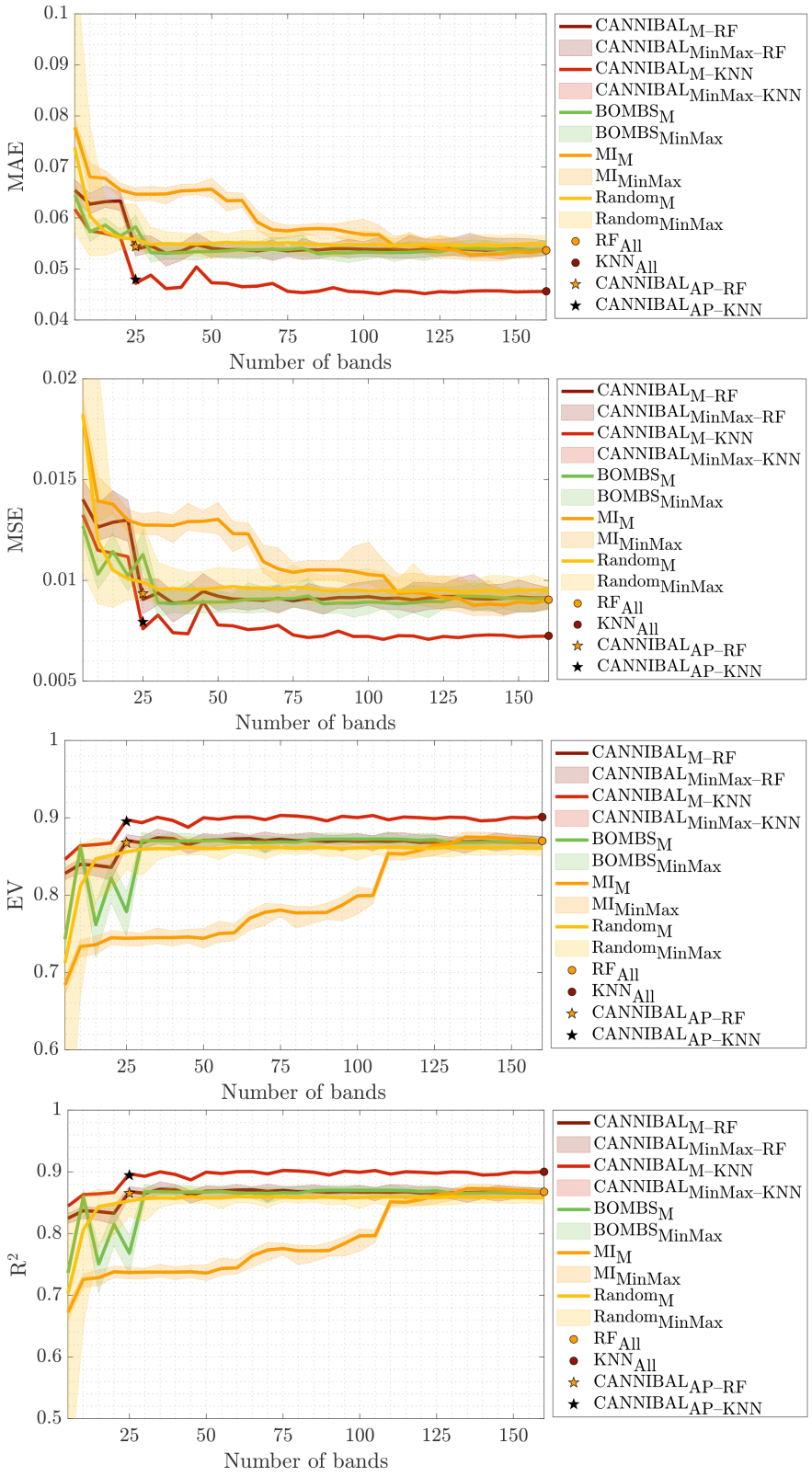


FIGURE 5.21: Quantification of the HU results, depicted as the MAE, MSE, EV, and R^2. This figure was utilized in the PhD Candidate's work in [240].

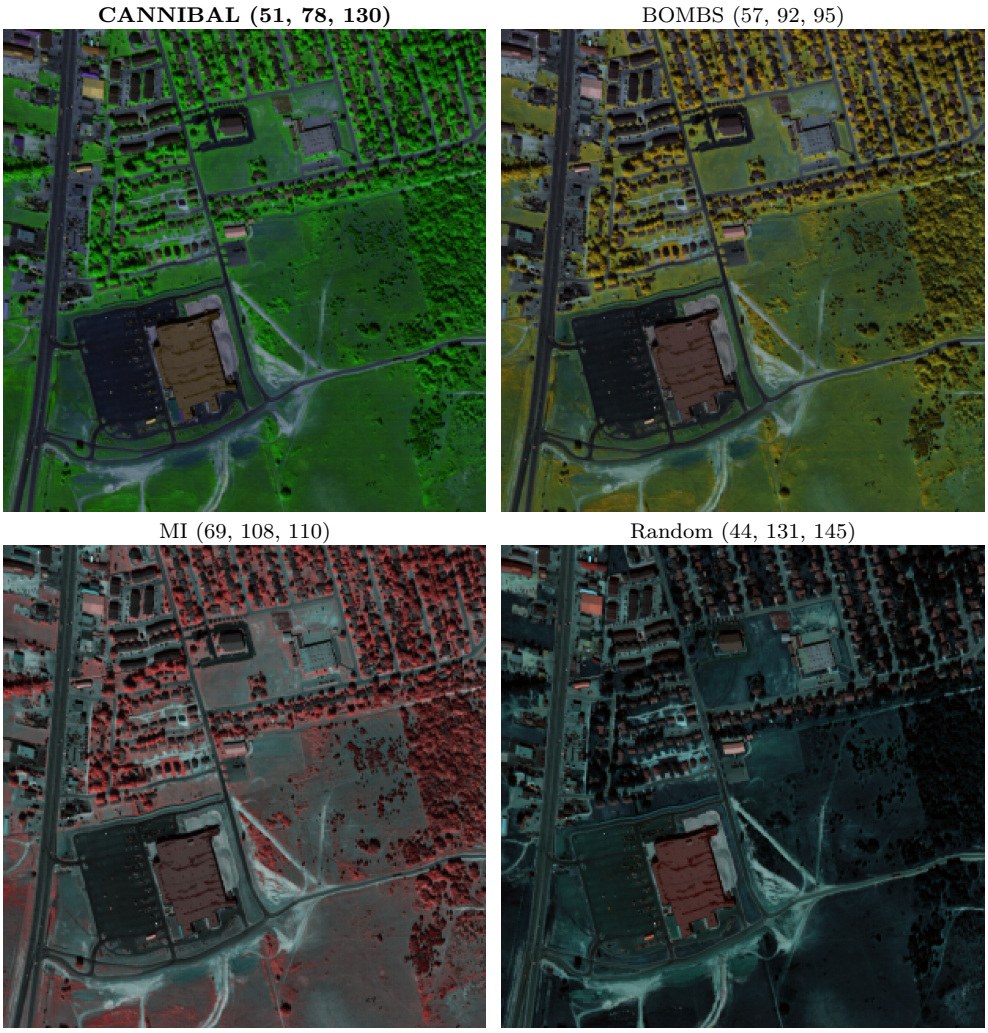


FIGURE 5.22: The RGB visualization of band selection process (the band identifiers are reported in parentheses) for all investigated algorithms for Urban benchmark dataset. This figure was utilized in the PhD Candidate's work in [240].

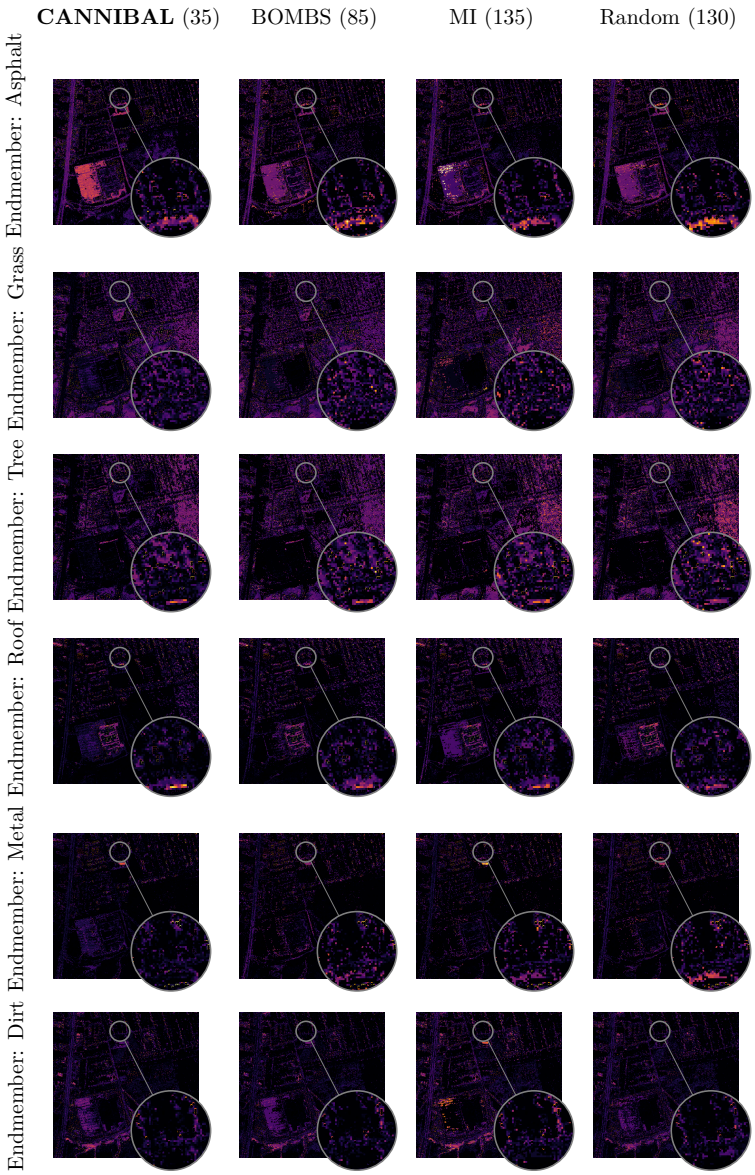


FIGURE 5.23: The HU errors for all endmembers are visualized for all the algorithms tested, using their optimal configurations, with the number of bands indicated in parentheses. Brighter colors represent higher error values. This figure was also utilized in the PhD Candidate's work in supplement of [240].

TABLE 5.13: The results of the Wilcoxon tests verify if the differences between the HU quality obtained by random forests trained on different numbers of bands selected by CANNIBAL (and trained over all bands) are statistically significant at $p < 0.05$. The background of statistically *insignificant* differences is rendered in light orange, and the corresponding p-values are boldfaced. The upper diagonal p-values were calculated for MAE, whereas the lower diagonal p-values were calculated for R^2. This table was also utilized in the PhD Candidate's work in supplement of [240].

$\mathcal{B}$	5	10	15	20	25	30	35	40	45	50	55	60	65	70	75	80	85	90	95	100	105	110	115	120	125	130	135	140	145	150	155	160	All
5		0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
10	0.000		0.022	0.036	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
15	0.000	**0.216**		**0.246**	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
20	0.000	0.040	**0.189**		0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
25	0.000	0.000	0.000	0.000		0.024	0.001	**0.083**	0.006	**0.498**	**0.812**	**0.701**	**0.123**	**0.898**	**0.143**	**0.622**	**0.409**	**0.522**	**0.546**	**0.956**	**0.898**	**0.430**	**0.143**	**0.729**	**0.596**	**0.812**	**0.546**	**0.546**	**0.756**	**0.812**	**0.522**	**0.898**	**0.546**
30	0.000	0.000	0.000	0.000	0.027		0.000	0.002	**0.784**	**0.058**	0.004	0.002	0.000	**0.143**	0.001	0.001	0.011	0.030	0.019	0.001	0.002	0.002	0.007	0.014	0.027	**0.070**	0.004	0.002	0.030	0.011	0.001	0.005	0.001
35	0.000	0.000	0.000	0.000	0.007	0.000		**0.231**	0.000	0.000	0.006	0.001	**0.105**	0.019	0.036	0.009	0.019	0.001	0.011	0.005	0.030	0.004	**0.231**	0.008	0.001	0.006	0.033	**0.064**	0.024	0.017	0.027	0.012	0.040
40	0.000	0.000	0.000	0.000	**0.083**	0.001	**0.927**		0.001	0.027	**0.294**	**0.312**	**0.546**	**0.368**	**0.812**	**0.546**	**0.498**	**0.105**	**0.123**	**0.154**	**0.294**	**0.105**	**0.622**	**0.388**	**0.133**	0.027	**0.312**	**0.756**	**0.177**	**0.189**	**0.312**	**0.090**	**0.498**
45	0.000	0.000	0.000	0.000	0.000	**0.165**	0.000	0.000		0.002	0.002	0.001	0.000	0.004	0.000	0.000	0.001	0.007	0.012	0.001	0.000	0.001	0.001	0.000	0.006	0.027	0.004	0.000	0.022	0.017	0.003	0.003	0.000
50	0.000	0.000	0.000	0.000	**0.701**	0.004	0.011	**0.133**	0.000		**0.294**	**0.294**	0.011	**0.522**	0.022	**0.177**	**0.189**	**0.648**	**0.784**	**0.452**	**0.294**	**0.231**	**0.202**	**0.388**	**0.927**	**0.522**	**0.522**	**0.105**	**0.622**	**0.756**	**0.409**	**0.452**	**0.083**
55	0.000	0.000	0.000	0.000	**0.522**	0.006	**0.053**	**0.165**	0.000	**0.985**		**0.841**	**0.261**	**1.000**	**0.498**	**0.648**	**0.622**	**0.674**	**0.596**	**0.784**	**0.756**	**1.000**	**0.231**	**0.729**	**0.648**	**0.571**	**0.701**	**0.246**	**0.870**	**0.956**	**0.784**	**0.784**	**0.368**
60	0.000	0.000	0.000	0.000	0.040	0.001	**0.330**	**0.648**	0.000	**0.202**	**0.430**		**0.261**	**0.956**	**0.133**	**0.898**	**0.870**	**0.409**	**0.784**	**0.985**	**0.870**	**0.674**	**0.261**	**0.985**	**0.498**	**0.596**	**0.927**	**0.596**	**0.674**	**0.674**	**0.812**	**0.841**	**0.498**
65	0.000	0.000	0.000	0.000	0.008	0.000	**0.522**	**0.870**	0.000	**0.083**	**0.114**	**0.622**		**0.154**	**0.956**	**0.498**	**0.430**	**0.053**	**0.053**	**0.097**	**0.277**	**0.090**	**0.701**	**0.277**	**0.114**	**0.246**	**0.294**	**0.571**	**0.154**	**0.123**	**0.261**	**0.053**	**0.312**
70	0.000	0.000	0.000	0.000	**1.000**	0.036	**0.058**	**0.177**	0.001	**0.596**	**0.841**	**0.277**	**0.189**		**0.123**	**0.648**	**0.349**	**0.812**	**0.622**	**0.571**	**0.870**	**0.898**	**0.409**	**0.674**	**0.648**	**0.870**	**0.870**	**0.498**	**0.729**	**0.898**	**0.701**	**1.000**	**0.368**
75	0.000	0.000	0.000	0.000	**0.114**	0.001	**0.177**	**0.409**	0.000	**0.277**	**0.596**	**0.841**	**0.368**	**0.261**		**0.622**	**0.430**	0.008	**0.154**	**0.097**	**0.622**	**0.277**	**0.812**	**0.277**	**0.064**	**0.070**	**0.349**	**0.498**	**0.177**	**0.123**	**0.430**	**0.165**	**0.571**
80	0.000	0.000	0.000	0.000	**0.388**	**0.070**	0.009	**0.070**	0.001	**0.701**	**0.388**	**0.064**	0.044	**0.546**	**0.097**		**0.841**	0.036	**0.133**	**0.546**	**0.956**	**0.546**	**0.522**	**0.330**	**0.154**	**0.430**	**0.956**	**0.812**	**0.522**	**0.596**	**1.000**	**0.701**	**0.784**
85	0.000	0.000	0.000	0.000	**0.870**	**0.083**	0.015	**0.114**	0.008	**0.898**	**0.522**	**0.216**	**0.083**	**0.956**	**0.202**	**0.571**		**0.231**	**0.312**	**0.870**	**0.596**	**0.648**	**0.648**	**0.452**	**0.165**	**0.475**	**0.870**	**0.756**	**0.841**	**0.674**	**0.870**	**0.729**	**0.648**
90	0.000	0.000	0.000	0.000	**0.312**	**0.294**	0.000	0.001	0.006	**0.076**	**0.083**	0.014	0.002	**0.202**	0.005	**0.409**	**0.312**		**0.841**	**0.546**	**0.312**	**0.756**	**0.143**	**0.349**	**0.898**	**0.985**	**0.430**	**0.064**	**0.898**	**0.596**	**0.216**	**0.452**	**0.097**
95	0.000	0.000	0.000	0.000	**0.927**	**0.064**	0.022	**0.076**	0.001	**0.648**	**0.596**	**0.202**	0.008	**0.784**	**0.177**	**0.729**	**0.571**	**0.312**		**0.546**	**0.330**	**0.246**	**0.388**	**0.546**	**0.927**	**0.985**	**0.452**	**0.261**	**0.812**	**0.927**	**0.294**	**0.648**	**0.177**
100	0.000	0.000	0.000	0.000	**0.349**	**0.143**	0.004	0.014	0.009	**0.409**	**0.189**	0.024	0.009	**0.143**	0.017	**0.648**	**0.294**	**1.000**	**0.522**		**0.927**	**0.729**	**0.330**	**0.927**	**0.522**	**0.674**	**0.622**	**0.277**	**0.927**	**0.756**	**0.622**	**0.898**	**0.216**
105	0.000	0.000	0.000	0.000	**0.571**	**0.133**	0.008	0.048	0.001	**0.330**	**0.231**	0.030	0.008	**0.294**	**0.123**	**0.927**	**0.312**	**0.596**	**0.475**	**0.898**		**0.956**	**0.452**	**0.596**	**0.498**	**0.784**	**0.870**	**0.701**	**0.956**	**0.985**	**0.674**	**0.870**	**0.571**
110	0.000	0.000	0.000	0.000	**0.330**	**0.349**	0.001	0.004	0.003	**0.053**	0.044	0.004	0.002	**0.105**	0.012	**0.330**	**0.246**	**0.898**	**0.261**	**0.729**	**0.622**		**0.596**	**0.729**	**0.756**	**0.756**	**0.648**	**0.349**	**0.898**	**0.898**	**0.812**	**0.784**	**0.475**
115	0.000	0.000	0.000	0.000	**0.956**	**0.154**	0.004	0.030	0.002	**0.756**	**0.312**	**0.202**	0.033	**0.409**	**0.154**	**0.985**	**0.571**	**0.475**	**0.927**	**0.956**	**0.841**	**0.312**		**0.349**	**0.165**	**0.409**	**0.701**	**1.000**	**0.674**	**0.622**	**0.648**	**0.522**	**0.927**
120	0.000	0.000	0.000	0.000	0.033	**0.812**	0.000	0.001	0.019	0.019	0.015	0.001	0.000	0.027	0.001	0.033	0.019	**0.349**	**0.070**	0.030	**0.114**	**0.231**	**0.058**		**0.571**	**0.841**	**0.898**	**0.231**	**0.674**	**0.956**	**0.985**	**0.898**	**0.622**
125	0.000	0.000	0.000	0.000	**0.133**	**0.522**	0.001	0.004	0.044	0.040	0.011	0.005	0.001	**0.133**	0.002	0.019	**0.058**	**0.956**	**0.143**	**0.430**	**0.294**	**0.546**	**0.177**	**0.812**		**0.985**	**0.312**	**0.114**	**0.898**	**0.430**	**0.231**	**0.571**	0.044
130	0.000	0.000	0.000	0.000	**0.143**	**0.729**	0.000	0.000	0.027	0.048	**0.090**	0.019	0.002	**0.143**	0.002	**0.246**	**0.105**	**0.571**	**0.202**	**0.409**	**0.368**	**0.622**	**0.294**	**0.571**	**0.985**		**0.522**	**0.189**	**0.956**	**0.927**	**0.330**	**0.622**	**0.133**
135	0.000	0.000	0.000	0.000	**0.165**	**0.231**	0.002	0.006	0.014	**0.090**	**0.083**	0.030	0.004	**0.154**	0.014	**0.123**	**0.165**	**0.898**	**0.165**	**0.571**	**0.522**	**0.729**	**0.452**	**0.475**	**0.841**	**0.756**		**0.349**	**0.927**	**0.648**	**0.674**	**0.841**	**0.475**
140	0.000	0.000	0.000	0.000	**0.349**	**0.154**	0.001	0.011	0.003	**0.114**	**0.090**	0.044	0.001	**0.143**	0.004	**0.349**	**0.246**	**0.756**	**0.154**	**0.870**	**0.674**	**0.898**	**0.546**	**0.123**	**0.330**	**0.546**	**0.388**		**0.261**	**0.294**	**0.812**	**0.349**	**0.985**
145	0.000	0.000	0.000	0.000	**0.154**	**0.475**	0.002	0.004	0.036	**0.105**	**0.097**	0.033	0.006	**0.133**	0.005	**0.165**	**0.154**	**0.571**	**0.154**	**0.546**	**0.622**	**0.701**	**0.841**	**0.388**	**0.756**	**0.985**	**0.701**	**0.246**		**0.756**	**0.622**	**0.701**	**0.261**
150	0.000	0.000	0.000	0.000	0.024	**0.571**	0.001	0.004	**0.058**	**0.090**	0.033	0.009	0.002	**0.123**	0.006	**0.165**	**0.143**	**0.498**	**0.165**	**0.216**	**0.231**	**0.475**	**0.430**	**0.729**	**0.701**	**0.729**	**0.956**	**0.312**	**0.729**		**0.756**	**0.956**	**0.058**
155	0.000	0.000	0.000	0.000	**0.165**	**0.189**	0.000	0.006	0.008	**0.154**	0.024	0.030	0.000	**0.189**	0.014	**0.246**	**0.123**	**0.985**	**0.143**	**0.571**	**0.452**	**0.898**	**0.277**	**0.452**	**0.784**	**0.784**	**0.898**	**0.622**	**0.898**	**0.927**		**0.701**	**0.409**
160	0.000	0.000	0.000	0.000	**0.123**	**0.388**	0.001	0.000	0.030	**0.053**	0.004	0.009	0.000	**0.165**	0.007	**0.133**	**0.123**	**0.409**	**0.053**	**0.368**	**0.368**	**0.522**	**0.177**	**0.648**	**0.841**	**0.870**	**0.729**	**0.294**	**0.841**	**0.870**	**0.956**		**0.312**
All	0.000	0.000	0.000	0.000	**0.674**	0.048	0.014	0.024	0.002	**0.498**	**0.143**	0.036	0.009	**0.729**	**0.076**	**0.956**	**0.648**	**0.522**	**0.674**	**0.784**	**0.898**	**0.648**	**0.898**	**0.070**	**0.076**	**0.202**	**0.261**	**0.729**	**0.277**	**0.090**	**0.143**	**0.053**	

In the Figure 5.22, we depict the visualizations of the Urban benchmark datasets for each of the investigated algorithms. As can be seen, all method provide a selection where distinct objects could be noticed. Nonetheless CANNIBAL allows us for more intuitive interpretation of the scene by capturing better color representation. Furthermore, in the Figure 5.23, we provide the visual representation of the unmixing errors for all investigated algorithms (horizontal axis) and all endmembers respectively (vertical axis). The brighter the color the higher the distance between the estimated abundance vectors and the ground-truth labels, hence higher unmixing error. It is clearly visible that CANNIBAL attributes much darker colors, and thus significantly lower errors when compared to other state-of-the-art band selection methods. Moreover, when investigating different areas over the Urban benchmark, we see much less anomalous regions, where the error spikes to large values for the CANNIBAL band selection approach.

5.4.6 Concluding Remarks

In this work, we proposed **CANNIBAL**, a novel, non-parametric band selection method designed for HSI analysis tasks such as unmixing and segmentation. By leveraging a clustering-based selection strategy, our approach dynamically identifies and excludes redundant spectral bands, thereby enhancing both the interpretability and computational efficiency of downstream processing pipelines.

Comprehensive experiments conducted on the widely utilized Urban HU benchmark dataset demonstrated that **CANNIBAL** consistently outperforms several state-of-the-art feature selection techniques across multiple performance metrics, including MSE, MAE, EV and (R^2). Notably, our method preserves high performance levels even when reducing the number of selected bands substantially, achieving statistically equivalent unmixing results to full-spectrum configurations when selecting as few as 25 bands, totaling up to $\tilde{8}5\%$ of data reduction.

An important characteristic of **CANNIBAL** is its ability to operate in a fully non-parametric manner, eliminating the need for user-defined parameters or thresholds. Additionally, the number of selected bands can be determined automatically through unsupervised clustering techniques, such as AP, without compromising the end result quality. Visual assessments of abundance estimations and unmixing errors further confirmed that **CANNIBAL** produces lower error rates and offers more consistent and intuitive scene representations compared to alternative methods.

Furthermore, pairwise Wilcoxon statistical tests validated the superiority of our approach in terms of performance stability and reliability across multiple initialization configurations. The results highlighted that the bands excluded through the "cannibalization" process indeed contained redundant or non-informative information, thereby substantiating the rationale behind our selection strategy.

It is critical for our work to emphasize that **CANNIBAL** is not limited to hyperspectral imaging and remote sensing applications. Owing to its data-driven, clustering-based formulation, the proposed methodology can be easily extrapolated

to other forms of multi-channel imagery, including biomedical imaging, industrial quality control systems, and multi-modal sensor data analysis. Its adaptability and scalability make it a promising candidate for feature selection in high-dimensional data domains beyond traditional HSI processing.

Our future work will focus on integrating **CANNIBAL** with advanced deep learning architectures and extending its application to end-to-end use-cases via automatic endmembers extraction.

5.5 Deep Ensembles for Hyperspectral Unmixing

In this work we focus on building heterogeneous ensemble models that incorporate as base learners cube-based CNNs that extract data from spectral as well as spatial domains of HSIs. The models are trained via supervised learning approach. Furthermore, we combine the multiple outputs of each learner and fuse them to produce the final output. The employed fusion methods can be selected and attributed based on the underlying problem. Our ensemble architecture allows for maximum flexibility, i.e., can incorporate any base model (not only DL-based methods). Moreover, our ensemble approach could be extrapolated to other tasks such as classification or even forecasting. To produce a set of robust learners, we employ noise augmentations of the base models. In this work, we utilize Gaussian distribution as the source of contamination that is injected to the trained convolutional-based model's weights. We hypothesize that such noise augmentation allows the model to obtain extra robustness to difficult conditions during HSI acquisition and greatly improves the generalization of the base models. Finally, our implementation is available at `https://github.com/ESA-PhiLab/hypernet/tree/master/beetles` and can be fully reproducible.

It should be noted that while the PhD Candidate's research has also explored the application of deep ensemble methods to hyperspectral segmentation tasks in [178], demonstrating the broader applicability of this ensemble framework, these segmentation experiments are not included in the current dissertation. Since the primary focus of this thesis is hyperspectral unmixing, the discussion and experimental results are deliberately concentrated on abundance estimation tasks to ensure comprehensive treatment and analytical depth within this specific domain. This focused approach allows for more thorough investigation of the ensemble methodology's performance in unmixing scenarios and provides clearer insights into its practical implications for real-world abundance estimation applications.

5.5.1 Proposed Fusing Method

In our approach, we utilize different fusing techniques. More specifically, we incorporate classical majority voting approach for segmentation, whereas for the unmixing task, we exploit the averaging aggregation. Furthermore, we exploit more advanced strategy of adding additional supervised models after the combination of multiple

features from the base learning. The supervised models are trained utilizing $\boldsymbol{o} \times N$ feature vectors, where $\boldsymbol{o}$ denotes the output feature vector of a single base learner. Conversely, N refers to the number of such learners. To provide an example, in the case of hyperspectral segmentation, the output of the base learners is a vector of the size equal to the number of classes, where each entry represents a target probability. In such case, the input to the fuser module incorporates N of such vectors $\mathcal{F} = [\boldsymbol{o}_1; \boldsymbol{o}_2; \ldots; \boldsymbol{o}_N]^T$, where *i-th* base model. Furthermore, the vectors later undergo majority (hard) voting or are fed to the subsequent supervised classifier.

To obtain the base learner, we train in a supervised manner a CNN model. Such model is later utilized for Gaussian noise injection. We incorporate those small perturbations directly in to the base learner's weights (after the training is completed). The probability density function that allows us to sample noise values can be formulated as:

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/2\sigma^2}, \tag{5.17}$$

where mean is denoted as μ, σ refers to standard deviation, and σ^2 is the variance parameter. For each layer of the CNN model, we inject the generated noise with zero mean and standard deviation set to $\sigma'_j = \epsilon \cdot \sigma_j^{\mathbf{w}}$, where the standard deviation of the kernels is denoted as $\sigma_j^{\mathbf{w}}$, ϵ is the scaler and can be treated as hyperparameter that measures the noise impact of the kernel's weights. We experimentally, chosen the value of $\epsilon = 0.1$. Nonetheless, the proper setting of this parameter can be further investigated in other problems and benchmarks. Finally, j constitutes the index of the underlying layer, where the noise will be injected. In the Figure 5.24, we provide a visual example of contamination of the model indexed by one and two. Such operation produced another set of models denoted with $1'$, $1''$, and $2'$ postfixes. The output of the added noise-augmented base learners is forwarded to the fuser, which later undergoes aggregation to form the final prediction.

In the case of hyperspectral image segmentation, we exploit three different architectures of CNN-based models. The specific topologies are defined in 5.1. The first model utilizes only the spectral dimension and it is referred to 1D-CNN. This architecture is build upon the model proposed in [177]. Additionally, we incorporate in the study two models named as 2.5-CNN [71] as well as 3D-CNN [182]. The following models utilize spectral-spatial dependencies to estimate the class of the central pixel, and the neighborhood shape for 2.5-CNN was suggested in [174]. The 2.5D-CNNs differ from 3D-CNNs in the mechanism of the convolutional layers. More specifically, the former variant extends all the kernels in the spectral dimensions, thus is limited in terms of feature extraction in this domain. Conversely, the 3D-CNN kernels move throughout all the axes (both spectral and spatial), thus are able to extract more relational information in those domains. We hypothesize that 3-dimensional convolution allows the model to enhance the quality of hyperspectral segmentation.

Furthermore, in the task of HU, we incorporate the architecture proposed in [281].

This model incorporates two versions, the spectral and spectral-spatial approaches. The Table 5.14 specifies the specific kernel dimensionalities of the both (pixel- and cube-based) variants.

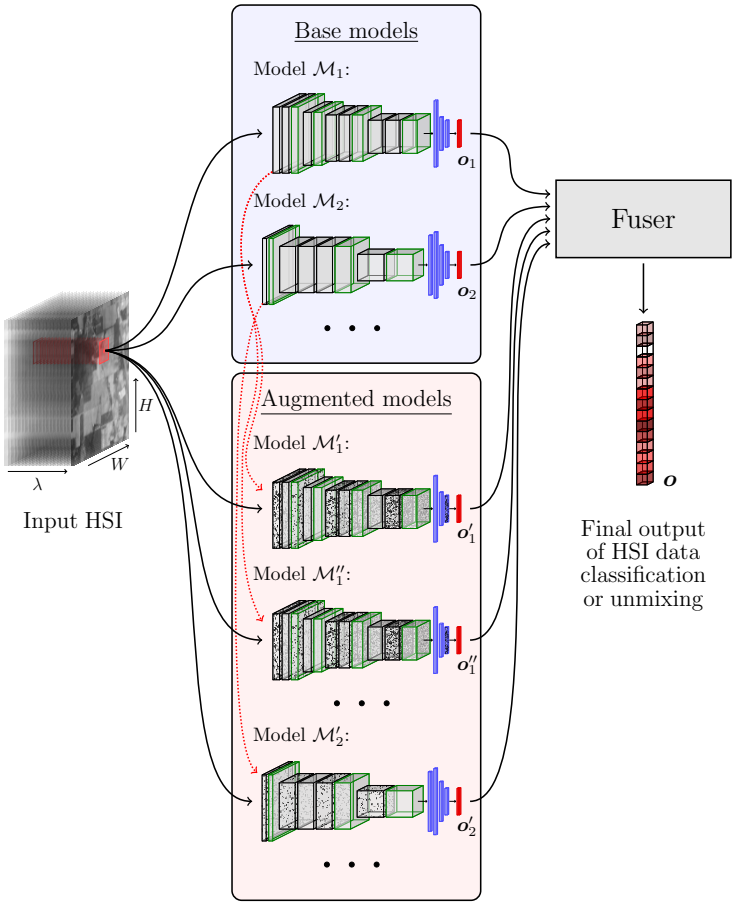


FIGURE 5.24: High-level view on the proposed deep ensemble-based approach in this work for HSI analysis. Each model can be a foundation of a set of noise-augmented variations. The noise perturbation follows the Gaussian distribution. The red arrows indicate the original (undisturbed) model that was utilized to create the further architecture. After all of the models perform the prediction, the fuser module combines the estimated abundance vectors (or in the case of segmentation class probabilities) and performs a aggregation. Of note, such aggregation is flexible and could incorporate nested logic and set of rules. Finally the final output is predicted. This figure was utilized in the PhD Candidate's work in [178].

TABLE 5.14: The CNN-based architectures utilized in the HU task. We note the number of filers, i.e., kernels as well as their dimensions. This table was utilized in the PhD Candidate's work in [178].

Variant	Layer	Parameters	Activation
1D-CNN $(1 \times 1 \times \lambda)$	Conv1	$3@1 \times 1 \times 5$	ReLU
	Conv2	$6@1 \times 1 \times 4$	ReLU
	Conv3	$12@1 \times 1 \times 5$	ReLU
	Conv4	$24@1 \times 1 \times 4$	ReLU
	FC1	$\# \times 192$	ReLU
	FC2	192×150	ReLU
	FC3	$150 \times a$	Softmax
3D-CNN $(3 \times 3 \times \lambda)$	Conv1	$16@1 \times 1 \times 5$	ReLU
	Conv2	$32@1 \times 1 \times 4$	ReLU
	Conv3	$64@1 \times 1 \times 5$	ReLU
	Conv4	$128@1 \times 1 \times 4$	ReLU
	FC1	$\# \times 192$	ReLU
	FC2	192×150	ReLU
	FC3	$150 \times a$	Softmax

TABLE 5.15: The DCAE model's topology. As in the previous Table 5.14, we report the number of utilized kernels together with their dimensionalities. This table was utilized in the PhD Candidate's work in [178].

Variant	Layer	Parameters	Activation
1D-DCAE $(1 \times 1 \times \lambda)$	Conv1	$2@1 \times 1 \times 3$	ReLU
	Conv2	$4@1 \times 1 \times 3$	ReLU
	Conv3	$8@1 \times 1 \times 3$	ReLU
	Conv4	$16@1 \times 1 \times 3$	ReLU
	Conv5	$32@1 \times 1 \times 3$	ReLU
	FC1	$\# \times 256$	ReLU
	FC2	$256 \times a$	+ Softmax
	FC3	$a \times \lambda$	ReLU
3D-DCAE $(5 \times 5 \times \lambda)$	Conv1	$16@3 \times 3 \times 3$	ReLU
	Conv2	$32@3 \times 3 \times 3$	ReLU
	Conv3	$64@1 \times 1 \times 3$	ReLU
	Conv4	$128@1 \times 1 \times 3$	ReLU
	FC1	$\# \times 256$	ReLU
	FC2	$256 \times a$	+ Softmax
	FC3	$a \times \lambda$	ReLU

5.5.2 Experimental Settings

As previously mentioned our experiments incorporate two underlying and fundamental tasks in the problem of HSI analysis, i.e., segmentation and unmixing. Our implementation was coded in `Python 3.6` and we utilize the `Tensorflow 1.12` package

for training the DL models. The experiments were conducted on NVIDIA Tesla T4 GPU.

In the HU experiments, we utilize the hyperparameters settings proposed in [114], [281]. More specifically, the spatial extend of the cube-based variant incorporates a 3×3 window. Additionally, we incorporate the formerly mentioned DCAE model [113], presented in the Table 5.15. The spatial neighborhood for this architecture was proposed in the original research paper, and we follow the recommendation, hence we exploit a 5×5 patch approach. In [113], the authors employ a autoencoder-based strategy, where the decoder constitutes a single fully-connected layer with its weights set to the endmembers matrix. In such a scenario, the DCAE's latent representation is extracted via the encoder part and composes the *code*. The model bases on the linear mixture model, which assumes that the spectral pixel vector is a linear combination of the estimated abundances and the endmembers matrix. Naturally, the number of endmembers has to be set *a priori* and is tied to the underlying HSI. It is important to emphasize that because of the fact that the DCAE is trained in a unsupervised manner, it cannot be incorporated in the supervised ensemble approach. Therefore, we validate the unmixing capabilities of this architecture by coupling multiple DCAEs trained over various training examples. To obtain the final prediction, we perform the averaging aggregation of their predictions. We also incorporate conventional machine learning methods as the baseline solutions. To this set we employ support vector regression (SVR) [33] as well as the linear mixing model that incorporates fully constrained least-squares (LMM) [88]. The hyperparameters for each approach was extracted via experiments and sensitivity analysis.

We train both of the DL-based models utilizing Adam optimizer with learning rates set to 0.001, 0.0001, 0.00001, and 0.00005 for the pixel- and cube-based forms of CNN and DCAE models respectively. As in the previous experiments the maximum number of epochs was set to 100 and the number of epochs without improvement conditioned on the validation loss, i.e., patience was set to 15 epochs. The validation set constitutes 10% of randomly selected pixels from the training data. Furthermore, the DCAE model incorporates the spectral information divergence (SID) loss proposed in [29]. Conversely for the CNN model, we utilize the MSE loss function.

To establish generic results and conclusions from the experiments, we repeat them 30 times following the Monte Carlo cross-validation split. Similarly as in the previous experiments, we sample 30 test sets that do not change with the change of the size of the training data.

5.5.3 Benchmark Datasets

In this study, we incorporate two HU benchmarks, i.e., the Jasper Ridge (JR) as well as Urban (Ur) datasets (Figure 5.25). The former one constitutes a 100×100 HSI with 198 hyperspectral bands. Furthermore, the Ur dataset contains a 307×307 scene that incorporates 162 channels. The JR benchmark employs four endmembers: *i*) road, *ii*) water, *iii*) soil, and *iv*) tree. In the Ur dataset, there are six endmembers with the

following materials: *i*) asphalt, *ii*) grass, *iii*) tree, *iv*) roof, *v*) metal, and *vi*) dirt. In both benchmarks, we exploit 7500 as well as 47000 samples in the training data for JR and Ur datasets respectively. In our study we investigate the influence of the varying size of the training data on the generalization over the test sets of all models. As in the previous research, we utilize $\{1, 6, 13, 33, 66\}\%$ variants approximately. Such training sets amount to $\{75, 500, 1000, 2500, 5000\}$ and $\{470, 2800, 6100, 15500, 31000\}$ training examples for JR and Ur benchmark datasets respectively.

5.5.4 Quality Metrics

To measure the performance of the unmixing algorithms, we utilize the root-mean square error (RMSE) as well as root-mean square abundance angle distance (rmsAAD) metrics. Such measures allow us to validate the distance between the estimated abundance fractions and the ground-truth labels and can be formulated as:

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{|\Psi|}(\boldsymbol{a}_i - \hat{\boldsymbol{a}}_i)^2}{|\Psi|}}, \tag{5.18}$$

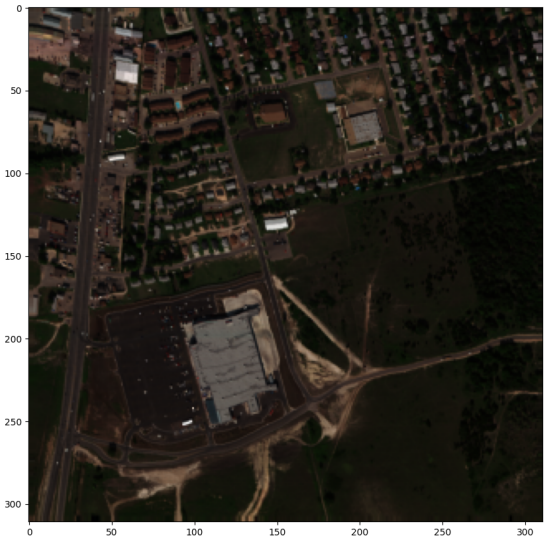
where $|\Psi|$ represents the total number of pixels in test set Ψ, whereas the $\boldsymbol{a}$ and $\hat{\boldsymbol{a}}$ vectors denote the ground-truth and estimated fractions of abundances. Furthermore, the rmsAAD can be formulated as:

$$\text{rmsAAD} = \sqrt{\frac{\sum_{i=1}^{|\Psi|} \arccos\left(\frac{\boldsymbol{a}_i^\top \hat{\boldsymbol{a}}_i}{\|\boldsymbol{a}_i\| \|\hat{\boldsymbol{a}}_i\|}\right)^2}{|\Psi|}}. \tag{5.19}$$

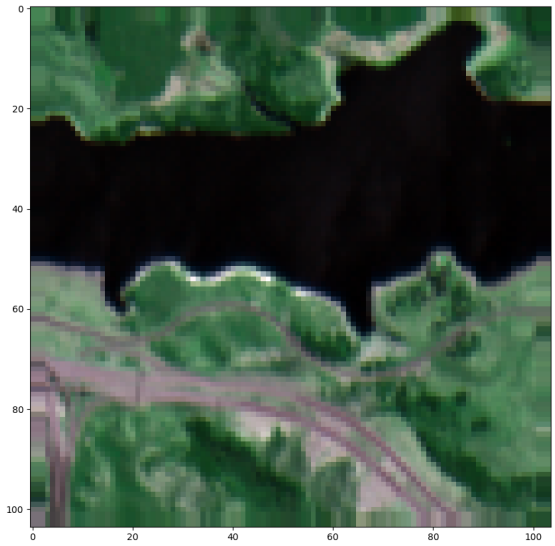
In the experiments where the fusers incorporate additional machine learning models (instead of standard aggregation or majority voting scenario), we utilize random forest (RF) regressor, with 100 decision trees (DT) as base estimators. Furthermore, we evaluate the DT and SVR fusers. The SVR incorporates a radial-basis kernel. We incorporate one such regressor model per endmember, as the SVR framework does not natively support multi-target problems.

5.5.5 Experimental Results and Discussion

The experimental results calculated for all variants of the training sets and methods selected in this work are depicted in the Figure 5.26. It is visible that when increasing the training set sizes, it allows the models to provide better abundances and over improves their generalization capabilities (as more examples are evident in the labeled data). Nonetheless, the difference between the larges and second-larges training set sizes are not significant and in fact, very little. This phenomenon can be observed for both CNN and DCAE models. Furthermore, when incorporating heterogeneous ensemble approach, it allows us to enhance the HU results for all training set sizes. It is important to emphasize that the RF fuser outperformed other fusing techniques (including the standard and conventional methods with averaging). The SVR approach offered the worst performance (probably due to separate regression



(A) Urban



(B) Jasper Ridge

FIGURE 5.25: Visual representation of the **(a)** Urban (Ur) and **(b)** Jasper Ridge (JR) datasets for the task of hyperspectral unmixing.

models for each endmember respectively). The ensemble approach also outperformed the classical methods, which incorporate LLM and SVR models. Of note, the SVR for the largest training set size failed to accomplish training during the maximum threshold of 12 hours (all methods were trained on the same device). In the Figure 5.27, we stipulate the training and test times for all investigated methods, averaged

across both unmixing benchmarks. The difference between standard and ensemble-based approach in terms of inference time is minimal, hence could be incorporated in production-ready environments.

Table 5.16 presents the performance results from noise augmentation experiments on HU, comparing single 1D-CNN models against ensembles of varying sizes (1-16 copies) using different fusion methods (Mean, RF, DT, SVR) on Urban and Jasper Ridge datasets. The results clearly demonstrate that ensemble-based modeling with noise augmentation significantly improves unmixing quality, with RF consistently achieving the best performance across both datasets—reducing RMSE from 6.76 to 6.40 for Urban and from 11.95 to 9.25 for Jasper Ridge when using 16 ensemble copies. The progressive improvement with increasing ensemble size suggests that weight perturbation through noise injection creates beneficial diversity among model predictions, leading to more robust and accurate abundance estimations. This approach proves particularly valuable for practical hyperspectral applications where ground-truth labeled data is scarce, as the noise augmentation strategy effectively expands the training diversity without requiring additional labeled samples, making CNN-based unmixing more feasible for real-world deployment scenarios.

Finally, Table 5.17 examines the computational performance of ensemble-based 1D-CNN models for HU, comparing training and testing (inference) times across different ensemble sizes (1-16 copies) and fusion methods (RF, DT, SVR, and mean-based) on Urban and Jasper Ridge datasets. The results show a clear trade-off between ensemble performance and computational cost, with training times increasing substantially as ensemble size grows—particularly evident in the Urban dataset where 16-copy ensembles require 3-4 times longer training. However, testing times remain reasonably manageable for practical deployment. Interestingly, mean-based fusion demonstrates competitive performance while maintaining lower computational overhead compared to more complex supervised methods like SVR. These findings suggest that while ensemble methods can enhance unmixing quality, researches must carefully balance ensemble size and fusion strategy based on their computational constraints and accuracy requirements, with mean-based fusion offering an attractive compromise for resource-limited applications.

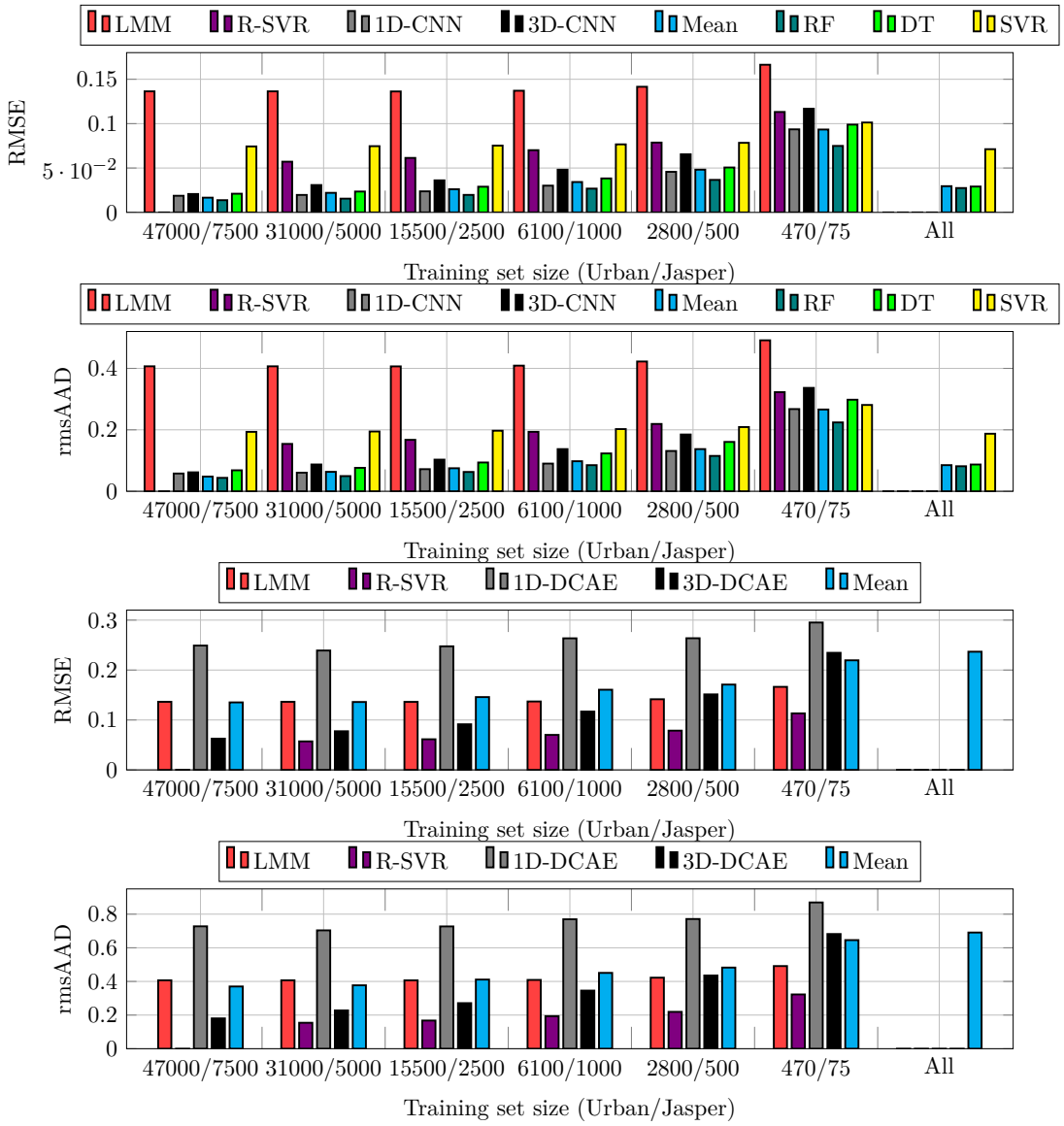


FIGURE 5.26: The performance outcomes—specifically, overall RMSE and rmsAAD averaged across Ur and JR—are presented for both 1D-CNN and 3D-CNN (displayed in the upper two plots), and for 1D-DCAE and 3D-DCAE (shown in the lower two plots). Additionally, results for ensemble approaches and traditional baseline methods, including LMM and R-SVR, are provided for comparison. For the supervised CNN models, RF, DT, and SVR are employed as fusion techniques. In both supervised and unsupervised scenarios, an ensemble strategy is also applied that computes the average prediction across base models (mean aggregation variant). For each training data size, heterogeneous ensembles are constructed combining both 1D and 3D model versions (CNNs and DCAEs). Furthermore, results are included for ensemble models that incorporate all base models trained on every evaluated training set size (referred to as the "Al" variant for both CNNs and DCAEs). It is noted that R-SVR results for the largest training sets are excluded, as the method did not complete training within the predefined 12-hour time limit. This table was utilized in the PhD Candidate's work in [178].

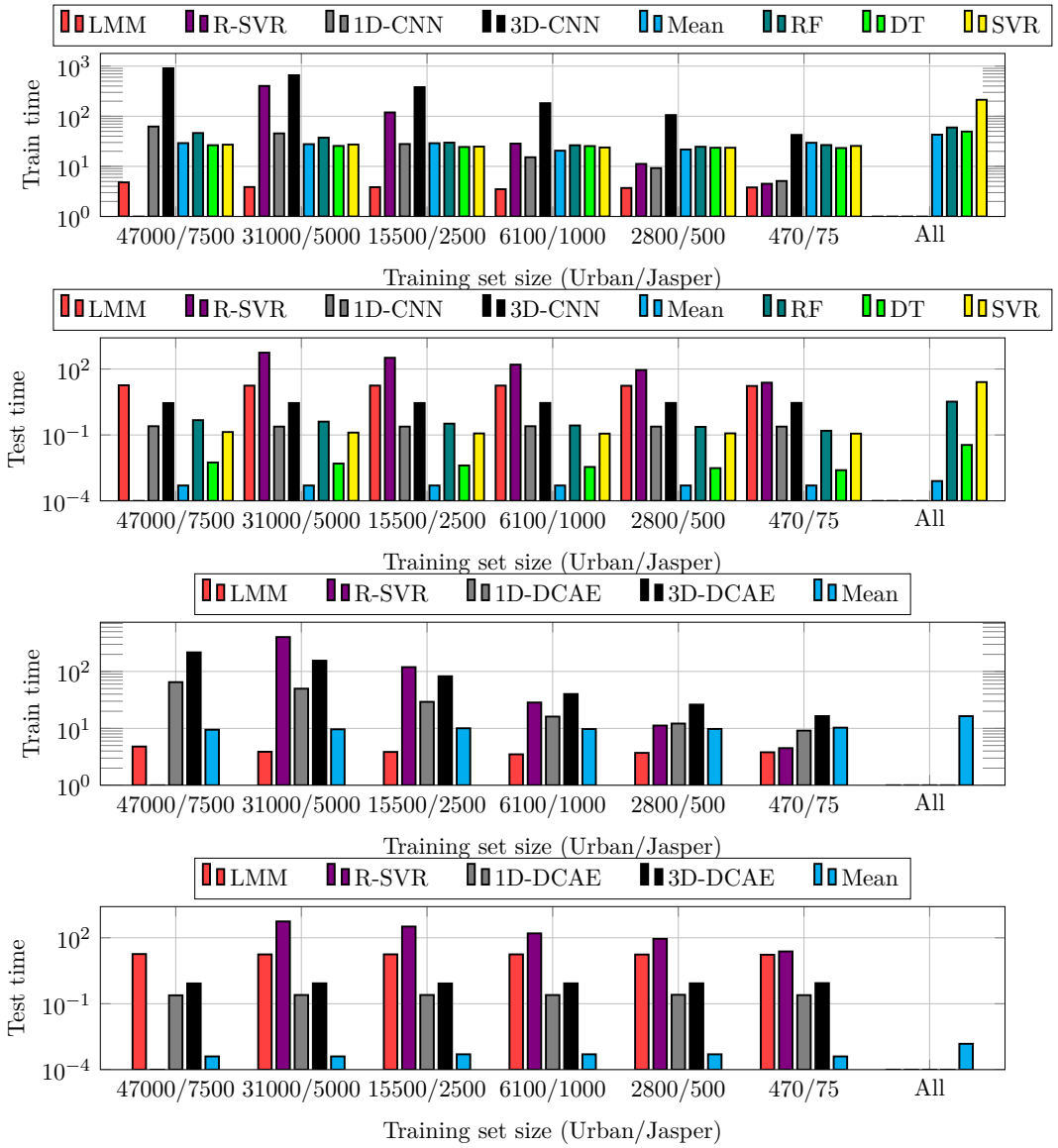


FIGURE 5.27: Training and prediction (testing) times, measured in seconds and presented on a logarithmic scale (note that test times correspond to the entire test sets), are provided for all considered training set sizes and algorithms, with values averaged across the datasets. The timing results for R-SVR on the largest training sets are omitted, as the method was unable to complete training within the designated 12-hour time limit. This table was utilized in the PhD Candidate's work in [178].

TABLE 5.16: The results shown in this table include those from a single 1D-CNN model (denoted as 1D) and from ensembles composed of varying numbers of augmented 1D-CNNs (# copies), using all fusion methods: Mean, RF, DT, and SVR. Each ensemble consistently includes the original 1D-CNN model. The augmented versions are produced by introducing noise into the original model's weights, with the number of these noise-perturbed (augmented) copies being adjustable as needed. Results are presented for both Ur and JR datasets. In each row, the best outcome is highlighted in bold, the second-best is underlined, and the poorest result is shown in gray. Both RMSE and rmsAAD values have been scaled by a factor of 100. This table was utilized in the PhD Candidate's work in [178].

	# copies→		1 copy				2 copies				4 copies				8 copies				16 copies			
	Metric↓	1D	Mean	RF	DT	SVR	Mean	RF	DT	SVR	Mean	RF	DT	SVR	Mean	RF	DT	SVR	Mean	RF	DT	SVR
Ur	RMSE	6.76	8.26	6.58	8.47	9.46	8.11	6.55	8.50	9.41	8.31	6.48	8.46	9.35	8.12	6.43	8.42	9.31	8.23	**6.40**	8.54	9.27
	rmsAAD	23.63	27.67	23.81	30.44	30.27	27.17	23.73	30.57	30.12	27.49	23.52	30.39	29.97	26.71	23.32	30.22	29.85	26.87	**23.22**	30.71	29.74
JR	RMSE	11.95	12.58	9.79	11.72	12.20	12.45	9.56	11.72	12.02	12.73	9.46	11.57	11.97	12.78	9.40	11.56	11.96	12.92	**9.25**	11.66	11.88
	rmsAAD	29.88	31.46	25.00	30.11	30.00	31.07	24.49	30.22	29.52	31.73	24.18	29.79	29.36	31.63	24.07	29.84	29.33	31.87	**23.72**	30.18	29.17

TABLE 5.17: Training and testing times (in seconds), with test times reported for the complete test sets, are provided for the evaluated ensembles comprising different numbers of 1D-CNN copies (# copies). The results cover all supervised fusion methods—RF, DT, SVR—as well as the mean-based fusion, and also include the baseline times for a single 1D-CNN model (1D). This table was utilized in the PhD Candidate's work in [178].

	# copies→		1 copy				2 copies				4 copies				8 copies				16 copies			
	Time.↓	1D	Mean	RF	DT	SVR	Mean	RF	DT	SVR	Mean	RF	DT	SVR	Mean	RF	DT	SVR	Mean	RF	DT	SVR
Ur	Train	6.21	4.20	5.89	5.58	9.87	4.56	5.25	4.79	4.86	5.09	6.12	5.99	5.63	20.29	8.43	8.21	7.72	12.19	16.83	14.44	13.95
	Test	0.39	0.37	0.56	0.26	0.55	0.57	0.69	0.40	0.75	0.97	0.94	0.64	1.17	1.98	1.49	1.18	2.03	5.06	2.48	2.27	3.81
JR	Train	4.01	3.60	3.94	4.52	4.95	3.89	4.65	4.80	5.61	4.57	5.65	4.97	5.82	6.16	7.13	7.95	6.98	10.03	13.66	13.22	12.99
	Test	0.09	0.15	0.06	0.04	0.04	0.22	0.07	0.05	0.05	0.40	0.08	0.06	0.08	0.90	0.12	0.10	0.11	2.93	0.19	0.16	0.17

5.5.6 Concluding Remarks

The comprehensive experimental evaluation of ensemble-based deep learning approaches for hyperspectral unmixing presented in this chapter yields several significant findings that advance the SOTA in HSI analysis. The experimental results demonstrate that heterogeneous ensemble approaches consistently outperform individual deep learning models across all investigated training set sizes. The integration of CNN and DCAE architectures through ensemble fusion techniques resulted in substantial improvements in both RMSE and rmsAAD metrics on the Jasper Ridge and Urban benchmark datasets.

The superior performance of the RF fuser compared to conventional averaging and SVR approaches highlights the importance of intelligent aggregation strategies in ensemble frameworks. The RF fuser's ability to learn complex non-linear relationships between base learner predictions and ground-truth abundances proves crucial for optimal ensemble performance, achieving the best results across both benchmark datasets. This finding demonstrates that the choice of fusion mechanism significantly impacts the overall ensemble effectiveness, with classical ML-based fusers outperforming simple averaging approaches.

The analysis of varying training set sizes reveals important insights into the generalization capabilities of deep learning models for hyperspectral unmixing. While increasing training data size from 1% to 33% and 66% of available samples consistently improves model performance, the marginal gains diminish significantly between the second-largest (66%) and third-largest (33%) training configurations. This phenomenon, observed across both CNN and DCAE architectures, suggests that these models reach a performance plateau with sufficient training data, indicating effective feature learning capabilities even with moderately-sized datasets. The ensemble approach demonstrates particular robustness across different training set sizes, maintaining superior performance even in data-scarce scenarios, which is especially valuable for practical HSI applications where ground-truth abundance data is expensive and time-consuming to obtain.

Subsequently, the timing analysis reveals that ensemble-based approaches maintain minimal difference in inference time compared to individual models without any additional overhead. The failure of SVR to complete training within the 12-hour threshold for the largest training set proves the computational advantages of the proposed deep learning ensemble framework. The fast prediction times achieved by the ensemble models demonstrate the practical viability of these approaches for real-world HSI processing applications where processing speed is critical.

The noise augmentation experiments provide compelling evidence that ensemble-based modeling coupled with synthetic noise injection significantly enhances unmixing quality. This finding is particularly significant for practical applications where HSI data often contains various forms of noise and artifacts. The improved robustness achieved through noise augmentation suggests that the proposed ensemble framework can effectively handle real-world data imperfections such as difficult atmospheric

conditions or noise, while maintaining high accuracy. The preservation of fast prediction times in augmented architectures ensures that the enhanced robustness does not compromise computational efficiency, making the approach suitable for operational deployment.

This work makes several important methodological contributions to HSI analysis, including the successful demonstration of heterogeneous ensemble integration combining supervised (CNN) and unsupervised (DCAE) architectures, validation of RF-based fusion as a superior aggregation strategy, and establishment of training set size requirements for optimal performance. The experimental results conclusively demonstrate that ensemble-based deep learning approaches represent a significant advancement in HU capabilities. The combination of improved accuracy, maintained computational efficiency, and enhanced robustness through noise augmentation positions these methods as viable solutions for operational HSI analysis applications, providing a solid foundation for further research into ensemble deep learning frameworks and offering promising directions for advancing automated HSI analysis.

5.6 Onboarding Deep Learning: Post-Training Quantization

In this chapter, we evaluate the DNNs' quantization capabilities coupled with real-life data. A comprehensive comparison between full- and limited-precision models is made that dives into qualitative and quantitative properties. Specifically, we examine how quantization affects model accuracy, latency, and memory footprint under practical deployment scenarios. We also explore the trade-offs between precision and performance, highlighting the conditions under which low-bit quantization (e.g., 8-bit representations) can be used without significant degradation in output quality. Furthermore, qualitative assessments are conducted using visualization techniques to better understand how quantization impacts feature representations and decision boundaries. This analysis not only demonstrates the effectiveness of post-training quantization (PTQ) but also provides insights into its applicability for edge and mobile computing environments.

5.6.1 Introduction to Quantization of Deep Learning Models for Hyperspectral Unmixing

In the field of DL, the quantization is a process where the full-precision models are reduced to much more compact representation. Usually, DNNs utilize 32-bit floating-point representation, which offers high precision but is computationally expensive and memory intensive. The quantization process allows us to replace the high-precision weight values with lower-precision alternative, such as 16-bit, 8-bit or even binary formats [42], [256]. This process is crucial in situations when deploying models on resource-constrained devices, e.g., smartphones [256], embedded systems, satellites [75], and Internet of Things hardware. It is important to emphasize that quantization may introduce a loss of accuracy of DNNs, nonetheless such techniques are necessary for certain conditions and there are also methods to alleviate such deviation.

One of the well-known techniques is PTQ [149]. It incorporates lowering the precision of weights of an already trained model. Also, it requires a dataset for calibrating the low bit quantization intervals of models' parameters and the input data. Recent research has focused on developing PTQ methods that eliminate the need for retraining or fine-tuning a model. Yoni et al. [42] introduced the Optimal Mean Squared Error technique, which aims to minimize the L2 distance between original and quantized tensors. Similarly, Ron et al. [13] proposed the Analytical Clipping for Integer Quantization method, which analytically determines the optimal clipping range and bit allocation per channel in neural networks. To address the issue of outlier channels, Zhao et al. [284] presented the Outlier Channel Splitting technique. Wang et al. [245] contributed a Bit-Split and Stitching framework to lower bit precision in PTQ and an Error Compensated Activation Quantization strategy to reduce quantization errors in activations. Nagel et al. [167] developed AdaRound, a data-

and task-aware weight-rounding method that formulates rounding as a quadratic unconstrained binary optimization problem via Taylor series approximation. In a more recent advancement, Nagel et al. [168] introduced Data-Free Quantization, which extends PTQ to zero-shot settings, where neither training nor testing data are accessible.

Furthermore, Quantization-Aware Training (QAT) [169], [190] is an advanced quantization approach that mimics low-precision computation during neural network training, allowing the model to adjust to quantized representations (e.g., 8-bit integers) without significant accuracy degradation. In contrast to PTQ, QAT introduces simulated quantization operations—including rounding and clipping effects—in the forward propagation step while retaining high-precision gradients during backpropagation. By exposing the model to quantization errors early in the training process, QAT enhances the robustness of weights and activations, making them more resilient to precision loss when deployed on edge devices or mobile platforms. This method often outperforms PTQ, particularly in low-bitwidth settings, as it optimizes both the model parameters and quantization ranges simultaneously. Popular DL frameworks such as TensorFlow and PyTorch support QAT through built-in tools, leveraging techniques like gradient straight-through estimators to handle the non-differentiable aspects of quantization.

5.6.2 Quantization Setup for Hyperspectral Unmixing

Field-Programmable Gate Arrays (FPGAs) are reconfigurable hardware platforms that provide parallel processing capabilities, low-latency execution, and energy-efficient operation, making them ideal for deploying deep neural networks in embedded and edge computing applications. This work explores the implementation of quantized neural networks using Microchip Technology Inc.'s VectorBlox Accelerator SDK [212], an open-source framework available on GitHub (`https://github.com/Microchip-Vectorblox/VectorBlox-SDK`). The VectorBlox ecosystem combines hardware and software components to accelerate neural network inference on FPGAs, particularly targeting the PolarFire FPGA with its integrated CoreVectorBlox IP core. The SDK offers a comprehensive workflow for model conversion, optimization, simulation, and deployment onto FPGA hardware.

To streamline this process, the VectorBlox SDK incorporates tools from Intel's OpenVINO Toolkit (2021.1 release), an open-source suite for deep learning inference optimization. Key integrated components include the OpenVINO Model Zoo, a model downloader, model converter to convert the model from various frameworks, and the Model Optimizer, which converts trained models from frameworks like TensorFlow and PyTorch into an intermediate representation compatible with VectorBlox:

- **OpenVINO Model Zoo** — it allows us access to multiple NNs architectures coupled with information about the preprocessing steps and performance. Furthermore, it also includes option to download the mentioned models.

- **OpenVINO Converter and Optimizer** — offers a mechanism to convert models from various DL frameworks into OpenVINO Intermediate Representation (IR). The optimizer allows us to remove training-time layers and operations, e.g., Dropout and applies layer fusion.

Afterwards, when the OpenVINO IR is created, the model is fed through the VectorBlox graph generation tool. The result of that is a quantized VNNX-extension based file that could be run on the simulator as well as real FPGA hardware. In this research, we focus only on the simulator. Together with the IR-based model, VectorBlox graph generation tool accepts a hardware configuration option, that may be set to V1000, V500, or V250. Those parameter values control the latency, degree of parallelism and computational resources used during inference process. For V250, a lower-performance is present with fewer resources used, resulting in lower inference speed. Such an option is recommended for smaller FPGAs. Furthermore, the V500 option provides higher throughput and increases the general speed of inference, thus is advocated for mid-range FPGAs. On the other hand, the V1000 option is best for compute-demanding real-time applications. It uses more memory and achieves lower latency to other possible options. We focus on examining the V1000 alternative. Lastly, the tool also accepts example images that are used during the quantization process. These images serve to calibrate the model weights by collecting activation statistics (e.g., min/max or histogram distributions) across the layers of the network. This data is essential for computing appropriate scaling and zero-point factors during quantization, ensuring the resulting integer-only model maintains accuracy close to the original floating-point model. The calibration process operates by propagating the representative dataset through the original floating-point model in inference mode, systematically recording the range of activations at each layer. For each layer l, the minimum and maximum activation values $(r_{min}^{(l)}, r_{max}^{(l)})$ are collected across all calibration samples. These statistics are then used to compute the quantization parameters: the scale factor, and the zero-point, which ensures that the floating-point zero maps exactly to an integer value. Both of the parameters are defined in [101]. The quality of calibration directly impacts the final model accuracy, as inadequate representative samples may lead to suboptimal quantization parameters that introduce significant HU numerical errors during inference. It is also worth mentioning that the user may optionally stop the graph at any point during the conversion process, producing a partial or truncated model after quantization. This feature is particularly useful when post-processing steps (such as decoding or bounding box filtering for object detection or instance segmentation) need to be performed externally, outside of the FPGA. By stopping the graph early, unnecessary operations are excluded from the quantized model, thereby reducing its size and complexity while enabling greater flexibility in deployment pipelines.

In selecting a suitable neural network model for quantization and the HU task, compatibility with both the VectorBlox Accelerator SDK and the OpenVINO toolkit was a central consideration. While the SDK supports a range of neural network layers

and architectures, not all models are equally compatible due to varying degrees of support for specific operations and input configurations.

The experimental evaluation was conducted using the VectorBlox SDK inference simulator, a functionally accurate simulation environment specifically designed for evaluating quantized neural networks on VectorBlox accelerator hardware. The VectorBlox SDK provides a comprehensive framework for compiling and executing quantized networks, supporting 64 different operators optimized for inference acceleration.

The VectorBlox simulator operates as a bit-accurate functional model of the VectorBlox accelerator, enabling precise performance evaluation without requiring physical hardware access. The simulation environment supports both C and Python application programming interfaces, providing flexibility in model evaluation and post-processing operations. For our experimental evaluation, we utilized the Python interface due to its enhanced data manipulation capabilities and streamlined integration with our analysis pipeline.

5.6.3 Experimental Settings: Neural Network Architecture Selection for Quantization in Hyperspectral Unmixing

During the initial evaluation phase, multiple CNN architectures were considered, including 1D, 2D, and 3D CNNs, all of which are relevant to HU tasks. 1D CNNs are effective at processing spectral data across a single dimension, while 3D CNNs simultaneously handle both spatial and spectral dimensions, offering a more comprehensive representation of hyperspectral information. Despite their theoretical advantages, both 1D and 3D CNNs posed significant compatibility challenges with the VectorBlox SDK. The primary issues encountered included unsupported layers, limitations in operation types, and difficulties in converting these models into the VNNX format required for deployment on the simulator and the VectorBlox hardware.

As a result of these constraints, the focus shifted to 2D CNN architectures, which proved to be fully compatible with the VectorBlox SDK. These models operate on two-dimensional spatial data and can accommodate hyperspectral information by treating spectral bands as additional input channels. This approach aligns well with the SDK's supported operations, enabling seamless model conversion and deployment. Moreover, 2D CNNs maintain sufficient modeling capacity to extract meaningful spatial and spectral features from hyperspectral imagery, making them a practical compromise between complexity and deployability.

By selecting 2D CNNs, the research ensures compatibility with the VectorBlox platform, thereby taking full advantage of its acceleration capabilities. This enables efficient, low-latency inference on simulator and possible FPGA hardware, which is particularly valuable in the space-constrained and performance-critical conditions. The decision to prioritize deployment feasibility over architectural complexity not only simplifies the possible development and integration process but also enhances the robustness and reliability of the system.

The neural network model used in this research is a CNN implemented using TensorFlow's Keras API. Designed for regression HU task on hyperspectral data, this model emphasizes both architectural effectiveness and deployment compatibility. Due to the potential for future adaptability, the model can be fine-tuned on new datasets if required. The full architecture of the model is shown in Figure 5.28 and is inspired and adapted from the PhD Candidate work in [233]. The model starts with three 2D convolutional layers that are incorporated to process the feature maps created from the input hyperspectral cube. The number of filters are 16, 32, and 64 respectively whilst, the kernel sizes are set to (1, 16), (1, 8), and finally (1, 4). We utilize unit stride in both dimensions. Afterwards, a flattening layer is incorporated to provide features that will undergo regression. The final part of the model constitutes a regressor, that utilizes two fully-connected layers with 64 and C units, respectively. The C variable denotes the number of endmembers in the final HU image. Assuming 6 endmembers and 162 channels, the total number of trainable parameters is equal to 574,262. Lastly, in the entire model, we apply ReLU [170] activation function to prevent from vanishing gradient problem [90]. Finally, the model was trained using the Adam [116] optimizer. Overall, this CNN architecture is tailored to effectively process hyperspectral data by leveraging multiple convolutional layers focused on spectral dimension extraction, followed by fully connected layers for accurate abundance regression for HU task.

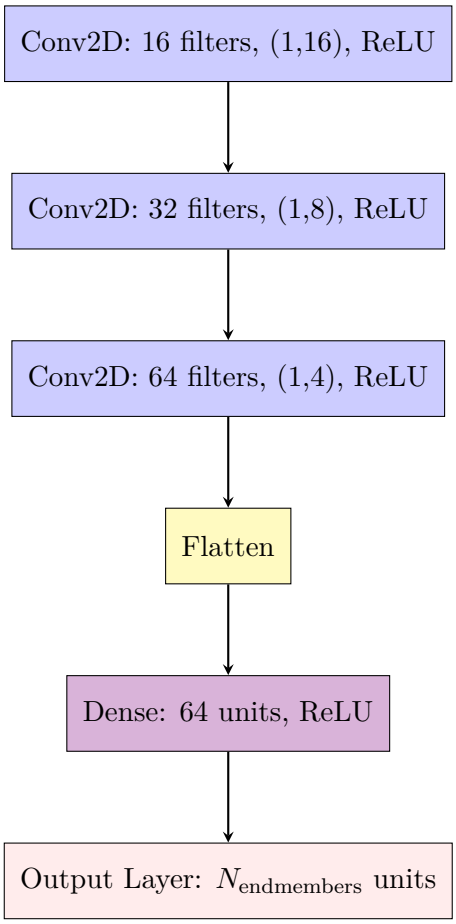


FIGURE 5.28: Block diagram of the CNN architecture used in the quantization experiment for HU.

5.6.4 Benchmark Datasets

To evaluate the performance of the proposed method, three widely used hyperspectral datasets were employed: Samson, Jasper Ridge, and Urban. These benchmarks provide ground truth data for both endmember spectra and abundance distributions.

The Samson dataset captures high-resolution spectral data from the visible to near-infrared range with approximately 3 nm spectral granularity. A representative $95 \times 95 \times 156$ region of interest was selected, containing three endmembers: *i*) soil, *ii*) tree, and *iii*) water.

The Jasper Ridge dataset spans the visible to shortwave infrared spectrum with moderately high spectral resolution. Noisy spectral bands were removed to ensure data quality. A $100 \times 100 \times 198$ region was selected, comprising four endmembers: *i*) road, *ii*) water, *iii*) soil, and *iv*) tree.

The Urban dataset covers various man-made and natural materials, with spectral data ranging from the visible to shortwave infrared regions at 10 nm intervals. Spectral bands affected by atmospheric effects or sensor noise were excluded. The selected $307 \times 307 \times 162$ subset includes six endmembers: *i*) asphalt, *ii*) grass, *iii*) tree, *iv*) roof, *v*) metal, and *vi*) dirt.

5.6.5 Quality Metrics

To effectively assess the performance of HU algorithms, it is important to rely on a set of well-defined evaluation metrics. These metrics help quantify the accuracy and reliability of abundance estimations, providing a basis for comparing different methods. Additionally, they are crucial for ensuring that the process of model quantization does not introduce significant performance degradation compared to the original floating-point implementation. In this work, we use the following standard metrics for evaluation:

- Mean Squared Error (MSE) quantifies the average squared difference between estimated and true abundance values:

$$\text{MSE} = \frac{1}{NP} \sum_{i=1}^{N} \sum_{j=1}^{P} \left(A_{ij} - \hat{A}_{ij} \right)^2 \tag{5.20}$$

  where A_{ij} and $\hat{A}_{ij}$ denote the ground truth and estimated abundance values of the j-th endmember at the i-th pixel, respectively. N is the number of pixels in the test set and P is the number of endmembers.

- Mean Absolute Error (MAE) measures the average absolute deviation between estimated and true abundances:

$$\text{MAE} = \frac{1}{NP} \sum_{i=1}^{N} \sum_{j=1}^{P} \left| A_{ij} - \hat{A}_{ij} \right| \tag{5.21}$$

  Similar to MSE, this metric uses absolute differences, making it less sensitive to large outliers while still reflecting overall prediction accuracy.

- Root Mean Squared Error (RMSE) is the square root of the MSE, offering a more stringent evaluation by penalizing larger errors:

$$\text{RMSE} = \sqrt{\frac{1}{NP} \sum_{i=1}^{N} \sum_{j=1}^{P} \left(A_{ij} - \hat{A}_{ij} \right)^2} \tag{5.22}$$

  RMSE retains the unit of the original data, making it more interpretable in the context of abundance fractions, while emphasizing larger deviations due to the squared term inside the root.

Together, these metrics form the basis for a reliable assessment of HU performance before and after quantization.

5.6.6 Experimental Results and Discussion: Onboarding Deep Learning Model via Quantization

In this subsection, we present the experimental results of HU measured on the VectorBlox inference simulator. The simulator is specifically designed for evaluating the

performance of quantized models. It's important to note that the values reported in the tables are presented with high numerical precision to capture any subtle differences, even though such results are typically shown with lower decimal precision in conventional reporting.

The results for the Urban dataset in Table 5.18 indicate that the **quantized model performs comparably to the full-precision model** across all evaluated metrics (MSE, MAE, and RMSE). Although the mean errors for MSE and RMSE are slightly higher in the quantized model, the differences remain minor, demonstrating only a slight decrease in accuracy. Additionally, the standard deviations and range between minimum and maximum values are similar for both models, suggesting consistent performance. This confirms that quantization preserves model accuracy well, making it a practical choice for scenarios with limited computational resources.

For the Jasper Ridge dataset depicted in Table 5.19, the quantized model shows error metrics close to those of the full-precision model. The mean values for MSE, MAE, and RMSE are very similar, with minor variations that fall within a comparable range. The error distribution, as evidenced by percentiles and standard deviation, further supports that quantization does not significantly affect predictive quality. This demonstrates the robustness of quantization for datasets with moderately large spectral dimensions.

In the Samson dataset (see Table 5.20), the quantized model exhibits slightly increased error metrics compared to the full-precision model, with marginally higher means and standard deviations for MSE, MAE, and RMSE. However, these differences are small and do not represent a significant degradation in performance. Both models share similar minimum and maximum values, and the error distributions suggest stable behavior. Therefore, quantization can be effectively applied to this dataset, offering a balance between model compression and accuracy.

Finally, the comparison of model sizes highlights the substantial storage savings achieved through quantization and model optimization. The full-precision model occupies approximately 25 MB of disk space. When converted to the OpenVINO IR format, the model size reduces to around 8.5 MB, representing a reduction of 66%. Further optimization with VNNX results in a compact model size of about 2.5 MB, **yielding an overall reduction of 90% from the original size** and an additional 71% reduction from the OpenVINO IR format. This corresponds to a size reduction ratio of approximately 3:1 when moving from full-precision to OpenVINO IR, and about 10:1 when using the VNNX format compared to the original full-precision model. Such significant compression enables more efficient storage and faster model loading, which are critical factors for deployment on satellite systems and other embedded platforms with limited memory and bandwidth.

TABLE 5.18: Experimental results obtained for Urban dataset for full-precision and quantized model over the test set. The metrics are reported over 4713 samples. Statistical values include Mean (average), Std (standard deviation), Min (minimum), 25% (first quartile), Median (second quartile), 75% (third quartile), and Max (maximum).

Model	Metric	Samples	Mean	Std	Min	25%	Median	75%	Max
Full	MSE	4713	0.0032982	0.001378	6.13E-07	0.0023421	0.0023175	0.003267	0.028059
Quantized	MSE	4713	0.0013287	0.001378	4.05E-06	0.0002691	0.0005967	0.001342	0.034089
Full	MAE	4713	0.0162193	0.011845	0.000591	0.0023185	0.0134827	0.020412	0.117701
Quantized	MAE	4713	0.0212841	0.013372	0.001443	0.0120382	0.0178694	0.023221	0.126675
Full	RMSE	4713	0.0223795	0.015799	0.000789	0.0120413	0.0223129	0.023239	0.167527
Quantized	RMSE	4713	0.0287658	0.018205	0.002014	0.0161893	0.0242467	0.036569	0.184657

TABLE 5.19: Experimental results obtained for Jasper Ridge dataset for full-precision and quantized model over the test set. The metrics are reported over 500 samples. Descriptions of the column headers are provided in Table 5.18.

Model	Metric	Samples	Mean	Std	Min	25%	Median	75%	Max
Full	MSE	500	0.0035987	0.004289	2.09E-06	0.0005312	0.0016935	0.0060153	0.051154
Quantized	MSE	500	0.0030812	0.003991	7.85E-07	0.0004261	0.0015198	0.0051524	0.052138
Full	MAE	500	0.0389342	0.025127	0.001132	0.0177991	0.0328659	0.0601382	0.177098
Quantized	MAE	500	0.0359186	0.024069	0.000648	0.0160183	0.0304196	0.0560849	0.179371
Full	RMSE	500	0.0501946	0.032526	0.001438	0.0229872	0.0411027	0.0774961	0.226179
Quantized	RMSE	500	0.0462239	0.030988	0.000886	0.0205523	0.0388868	0.0717221	0.228349

TABLE 5.20: Experimental results obtained for Samson dataset for full-precision and quantized model over the test set. The metrics are reported over 452 samples. Descriptions of the column headers are provided in Table 5.18.

Model	Metric	Samples	Mean	Std	Min	25%	Median	75%	Max
Full	MSE	452	0.0074893	0.012078	1.34E-07	0.0008472	0.0027784	0.0118752	0.138169
Quantized	MSE	452	0.0102231	0.015283	5.47E-07	0.0009913	0.0033928	0.0158811	0.139599
Full	MAE	452	0.0578856	0.046417	0.000301	0.0246827	0.0434542	0.0892016	0.350381
Quantized	MAE	452	0.0668517	0.054069	0.000606	0.0272613	0.0482215	0.1066078	0.352211
Full	RMSE	452	0.0683191	0.053146	0.000369	0.0290051	0.0526733	0.1089362	0.371713
Quantized	RMSE	452	0.0791885	0.063001	0.000742	0.0314182	0.0581729	0.1259734	0.373632

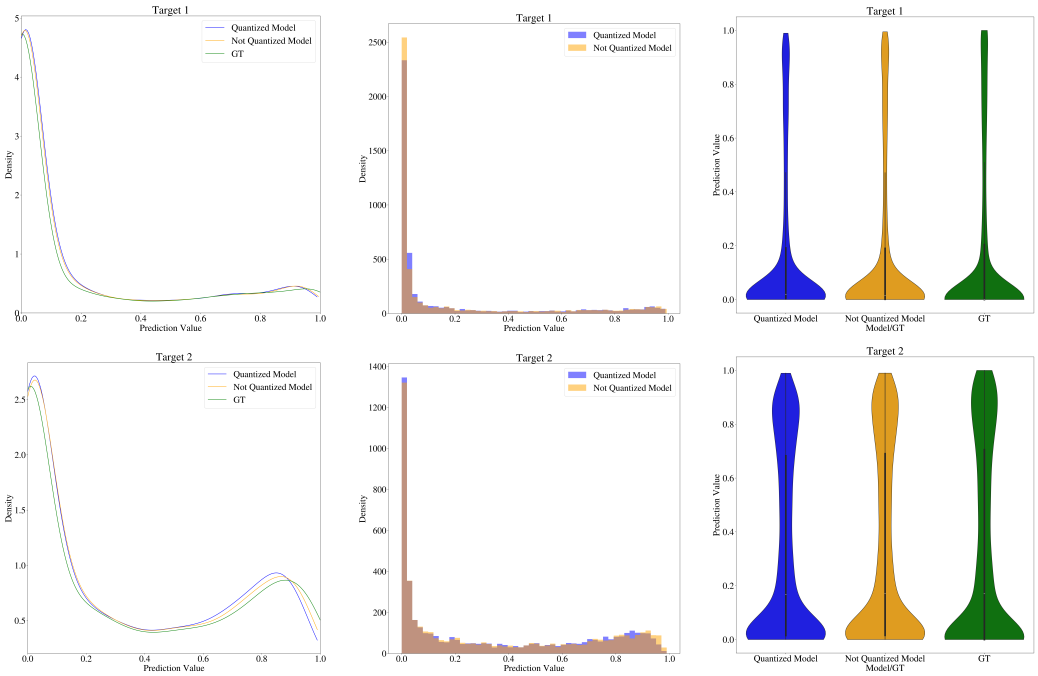


FIGURE 5.29: Distribution plots (KDE, Histogram, Violin) for Targets 1 and 2 for Urban dataset.

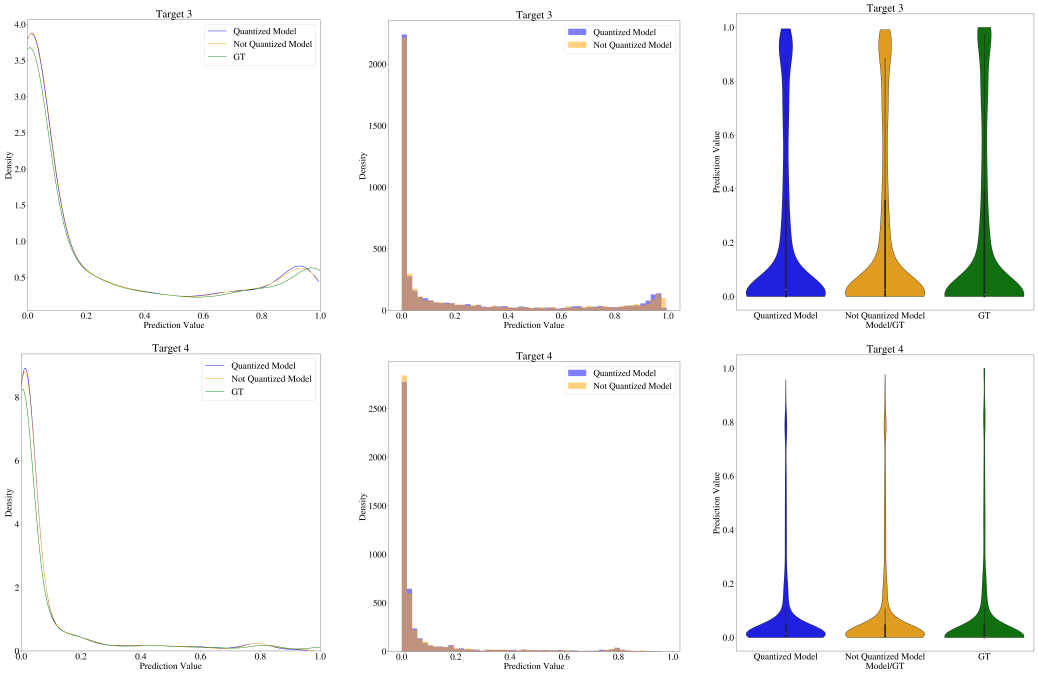


FIGURE 5.30: Distribution plots (KDE, Histogram, Violin) for Targets 3 and 4 for Urban dataset.

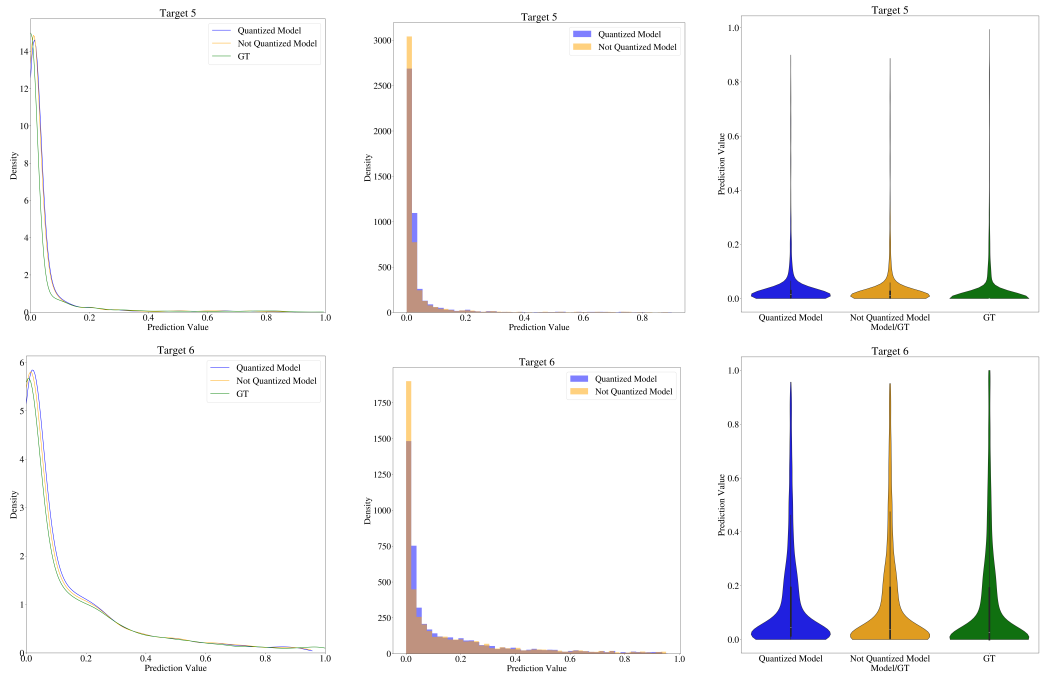


FIGURE 5.31: Distribution plots (KDE, Histogram, Violin) for Targets 5 and 6 for Urban dataset.

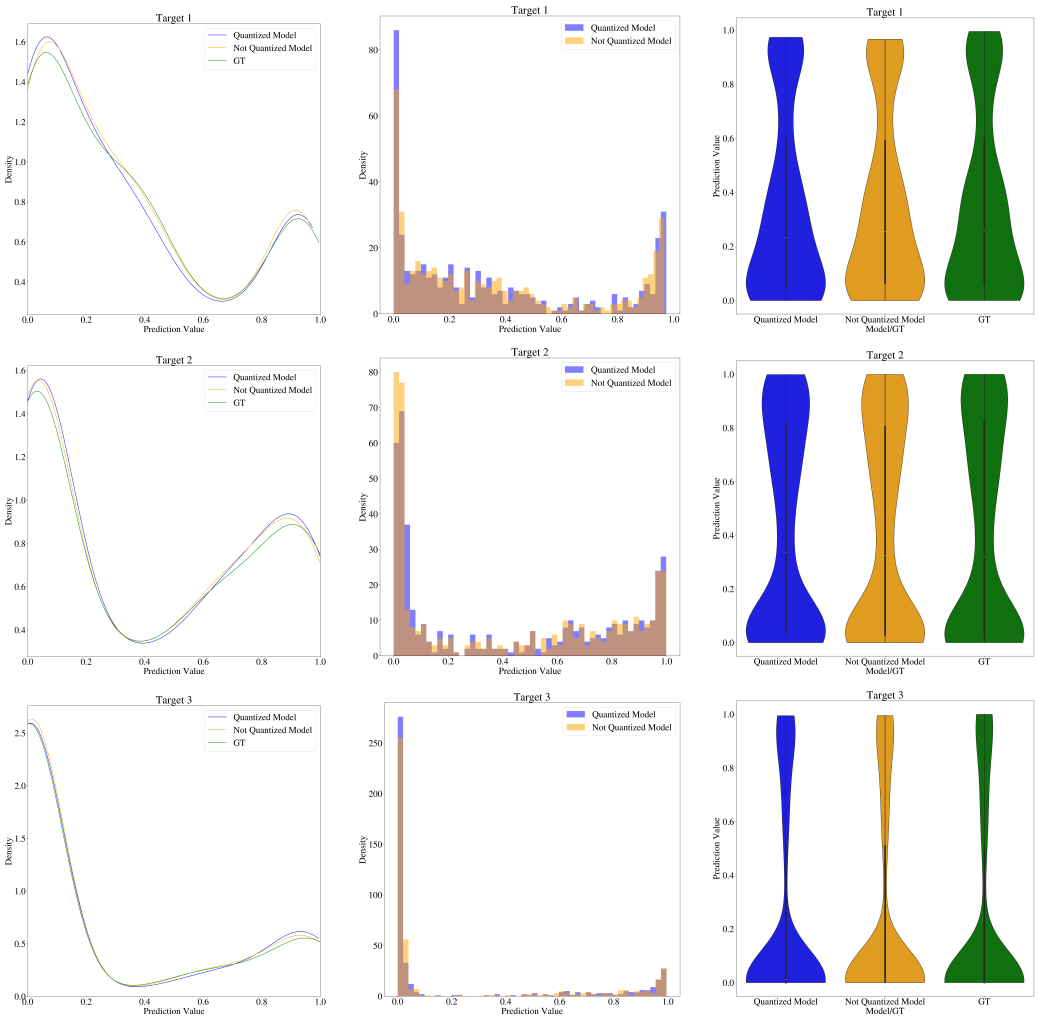


FIGURE 5.32: Prediction distributions for all 3 targets from the Samson dataset. Each row corresponds to one target, with the kernel density estimation (KDE) plot (left), histogram (middle), and violin plot (right) comparing predicted and ground truth soft labels using quantized and non-quantized models.

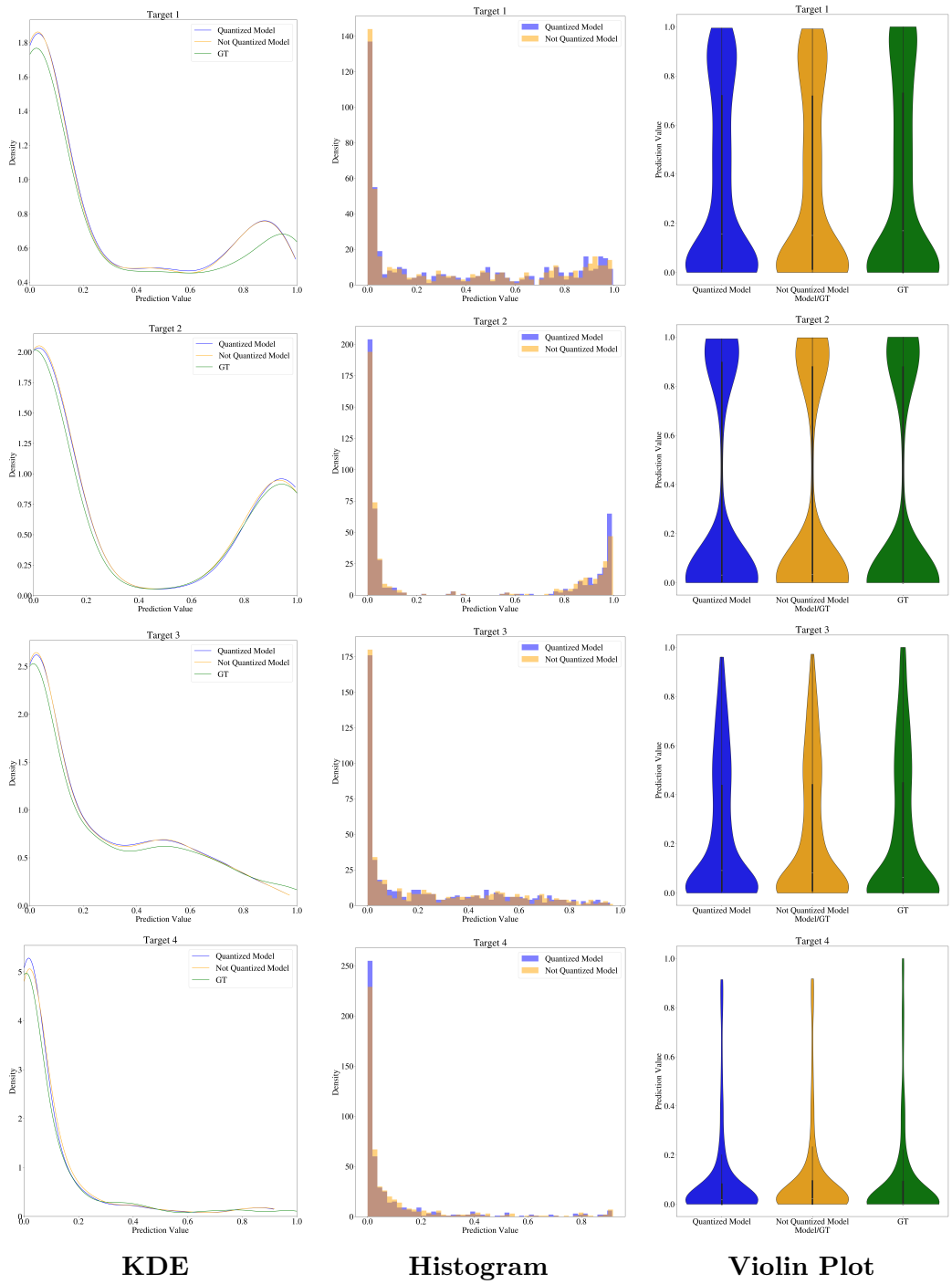


FIGURE 5.33: Prediction distributions for all 4 targets from the Jasper Ridge dataset. Each row corresponds to one target, with the kernel density estimation (KDE) plot (left), histogram (middle), and violin plot (right) comparing predicted and ground truth soft labels using quantized and non-quantized models.

Across all three hyperspectral datasets, the quantized models maintain performance close to their full-precision counterparts. The slight differences in error metrics indicate that quantization is an effective technique to reduce model complexity and memory demands while preserving reliable prediction accuracy. This makes quantization especially useful for real-time or resource-constrained applications [252].

Figures 5.29, 5.30, 5.31, 5.32, and 5.33 presented above, provide a detailed comparison of predicted soft label distributions from quantized and non-quantized models against the ground truth data for multiple targets across different datasets. For each target, three types of plots are included:

- **KDE (Kernel Density Estimation)** plots show smooth estimates of the probability density functions of predicted and ground truth soft labels, facilitating the visualization of distribution shapes and overlaps.
- **Histogram** plots display the frequency distribution of predicted and ground truth values in discrete bins, offering insight into the common value ranges.
- **Violin plots** combine box plot statistics with KDE to illustrate the shape, spread, and density of distributions, enabling direct comparison among quantized predictions, non-quantized predictions, and ground truth.

Together, these visualizations illustrate how effectively the models approximate the true soft label distributions for each target, shedding light on both the similarities and differences across the Urban, Samson, and Jasper Ridge datasets. Notably, the quantized model demonstrates a strong alignment with the full-precision model, maintaining nearly identical output distributions across all endmembers in each benchmark dataset. This consistency suggests that quantization does not significantly degrade the model's ability to estimate abundance maps with high fidelity. In fact, the preserved distributional characteristics indicate that the quantized model retains the essential representational capacity needed for accurate soft label prediction. As a result, it offers a compelling balance between computational efficiency and predictive performance, particularly in resource-constrained deployment scenarios.

5.6.7 Concluding Remarks

This work evaluated the impact of model quantization on HU performance across three widely used datasets: Urban, Jasper Ridge, and Samson. Both full-precision and quantized versions of the model were tested on identical test sets, allowing for a direct comparison of accuracy metrics such as MSE, MAE, and RMSE.

The results consistently show that the quantized model achieves performance comparable to the full-precision counterpart. For the Urban dataset, quantization led to only minor increases in error values, with mean MSE, MAE, and RMSE remaining closely aligned between models. Similarly, the Jasper Ridge dataset results indicate negligible accuracy loss after quantization, with error distributions nearly overlapping. The Samson dataset, which exhibits more variability, showed slightly higher errors in the quantized model but still maintained overall stable and acceptable performance.

These findings demonstrate that quantization can significantly reduce model size and computational demand without substantially sacrificing predictive accuracy in HU tasks. The minor increases in error observed are outweighed by the efficiency

benefits, making quantized models a practical choice for deployment in resource-constrained environments. In particular, the reduced memory footprint and computational requirements of the quantized models open up promising possibilities for deployment directly on satellites or other embedded platforms where hardware resources, power consumption, and bandwidth are highly limited. Real-time or near-real-time hyperspectral analysis onboard satellites can enable faster data processing and decision-making, minimizing the need for extensive data transmission to ground stations. This capability is crucial for applications such as environmental monitoring, disaster response, and precision agriculture. Overall, the experiments validate quantization as a robust approach for optimizing hyperspectral unmixing models, balancing efficiency and accuracy effectively across diverse datasets and spectral characteristics.

Chapter 6

Conclusions

This dissertation presents a comprehensive investigation into HU through multiple interconnected research gaps, addressing critical challenges in training optimization, validation methodology, architectural efficiency, feature selection, ensemble learning, and model deployment. The collective findings establish a robust framework for practical and reliable hyperspectral analysis across diverse operational and resource-constrained scenarios.

6.1 Training Set Optimization and Validation Methodology

Our investigation into training set size effects revealed that abundance estimation quality deteriorates significantly for smaller training sets, particularly those comprising less than 6% of the HSI. Importantly, we demonstrated that increasing training set sizes beyond 33% yields no statistically significant improvements, establishing a practical threshold for optimal data utilization. This finding has profound implications for operational efficiency, as it indicates that extensive labeling efforts beyond this threshold provide diminishing returns.

Equally critical is our development of an unbiased validation methodology that addresses the frequent issue of information leakage due to spatial correlations in HSI data. Our proposed non-overlapping sampling algorithm revealed that traditional random sampling methods overestimate algorithm performance by approximately 550%, fundamentally compromising the reliability of model evaluation in the literature that utilizes random sampling approach. The algorithm's parameterizable nature, multi-fold cross-validation capability, and support for both pixel-wise and patch-wise sampling establish it as an essential tool for rigorous hyperspectral algorithm validation. This methodological contribution ensures that model generalization capabilities are accurately assessed, which is crucial for operational deployment and safety-critical applications.

6.2 Architectural Innovation and Efficiency

Our comprehensive evaluation of deep learning architectures for HU demonstrates that different approaches serve distinct operational requirements. Multi-branch architectures, leveraging spectral and spectral-spatial feature fusion through 1D, 2D, and 3D convolutional branches, continue to achieve superior abundance estimation performance with statistically significant improvements in RMSE and rmsAAD metrics. However, our ablation studies revealed that the combination of 1D and 3D branches provides the most substantial benefits, while 2D branches offer limited additional value due to redundancy and insufficient spatial discrimination.

Most notably, our investigation of GCNNs and their Chebyshev variants revealed a paradigm shift toward efficiency without substantial performance compromise. The parameter efficiency is particularly striking: GCNN requires only 363,235 trainable parameters while GCNN-Cheb uses 620,071 parameters, representing dramatic reductions compared to state-of-the-art alternatives. For context, the multi-branch (MB) architecture requires 68,079,974 parameters—over 187 times more than GCNN and 110 times more than GCNN-Cheb. Even compared to other efficient architectures, GCNNs demonstrate superior parameter efficiency: they require approximately 3 times fewer parameters than CB-CNN [280] (91,006,624), 5 times fewer than UnDIP [201] (1,738,970), and 14 times fewer than WS-AE [11] (5,094,420). Despite this substantial parameter reduction, GCNNs achieve performance comparable to these computationally intensive models while demonstrating superior training efficiency and rapid convergence. This exceptional efficiency, combined with their inherent interpretability advantages for explainable AI approaches, positions GCNNs as particularly attractive for resource-constrained deployment scenarios, including satellite-based applications where memory and computational resources are severely limited.

6.3 Feature Selection and Dimensionality Optimization

The development of CANNIBAL as a non-parametric feature selection method addresses the critical challenge of hyperspectral data dimensionality. Our approach automatically determines optimal band selection through unsupervised clustering, achieving effective dimensionality reduction while maintaining high-quality unmixing performance without significant score deterioration. Statistical analysis through pairwise Wilcoxon tests over benchmark datasets confirmed that configurations with more than 25 CANNIBAL-selected bands perform statistically equivalent to full-spectrum analysis (in the utilized benchmark dataset the total number of bands amounts to 162, thus allowing us for 85% data reduction), validating the elimination of redundant spectral information. Furthermore, the method's ability to consistently outperform other feature selection approaches with statistical significance, while requiring significantly fewer features, demonstrates its practical value for computational efficiency

and storage optimization in operational systems.

6.4 Ensemble Learning and Data Augmentation

Our investigation into ensemble-based modeling coupled with noise augmentation revealed substantial potential for enhancing HU quality, particularly in scenarios with sparse and limited ground-truth labeled data. The ensemble approach maintains rapid prediction times while improving robustness, making it especially valuable for real-world applications where training data availability is constrained. This contribution is particularly relevant for accelerating the adoption of CNN-based unmixing models in operational scenarios.

6.5 Model Quantization and Practical Deployment

The evaluation of post-training quantization across multiple benchmark datasets (Urban, Jasper Ridge, and Samson) demonstrates that model compression can be achieved without substantial accuracy sacrifice. Quantized models maintain performance closely aligned with full-precision counterparts while significantly reducing computational demands and memory footprints up to 90% of overall reduction from the original full-precision model. This capability opens promising avenues for direct deployment on satellites and embedded platforms where hardware resources, power consumption, and bandwidth are severely constrained.

6.6 Integrated Framework and Future Directions

Collectively, this research establishes a comprehensive framework that balances data validation, accuracy, efficiency, and practical deployment considerations for HU. The integration of rigorous validation methodologies with efficient architectural designs, intelligent feature selection, ensemble approaches, and quantization techniques provides multiple pathways for optimizing HU systems according to specific operational requirements.

Key recommendations emerging from this work include: *i*) adoption of non-overlapping validation strategies to ensure reliable model evaluation, *ii*) utilization of training sets comprising 6-33% of available data for optimal efficiency-accuracy balance, *iii*) consideration of GCNN-based architectures for resource-constrained deployments, *iv* implementation of CANNIBAL feature selection for dimensionality optimization, and *v*) application of quantization techniques for edge deployment scenarios with high reduction rates.

Future research directions should focus on extending GCNNs architectural capabilities to further close performance gaps with complex state-of-the-art models, investigating transfer learning approaches to leverage the established training set thresholds across different domains, and developing adaptive ensemble strategies that

dynamically adjust to varying data availability and quality conditions. Additionally, the integration of the proposed methodologies into end-to-end hyperspectral processing pipelines for operational satellite missions represents a critical next step toward practical implementation.

This dissertation contributes both methodological approaches and practical solutions to the HU domain, establishing foundations for more reliable, efficient, and deployable hyperspectral analysis systems that can meet the demanding requirements of present-day remote sensing applications.

Bibliography

[1] M. Agarla, S. Bianco, L. Celona, R. Schettini, and M. Tchobanou, "An analysis of spectral similarity measures," in *Color and Imaging Conference*, Society for Imaging Science and Technology, vol. 29, 2021, pp. 300–305.

[2] A. H. K. Ahmadi, *Memory-based graph networks*. University of Toronto (Canada), 2020.

[3] F. I. Alam, J. Zhou, A. W.-C. Liew, and X. Jia, "Crf learning with cnn features for hyperspectral image segmentation," in *2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, IEEE, 2016, pp. 6890–6893.

[4] U. Alon and E. Yahav, "On the bottleneck of graph neural networks and its practical implications," *arXiv preprint arXiv:2006.05205*, 2020.

[5] J. M. Amigo, H. Babamoradi, and S. Elcoroaristizabal, "Hyperspectral image analysis. a tutorial," *Analytica chimica acta*, vol. 896, pp. 34–51, 2015.

[6] O. O. Aremu, D. Hyland-Wood, and P. R. McAree, "A machine learning approach to circumventing the curse of dimensionality in discontinuous time series machine data," *Reliability Engineering & System Safety*, vol. 195, p. 106 706, 2020.

[7] N. Audebert, B. Le Saux, and S. Lefèvre, "Deep learning for classification of hyperspectral data: A comparative review," *IEEE geoscience and remote sensing magazine*, vol. 7, no. 2, pp. 159–173, 2019.

[8] N. Audebert, B. Le Saux, and S. Lefèvre, "Deep learning for classification of hyperspectral data: A comparative review," *IEEE geoscience and remote sensing magazine*, vol. 7, no. 2, pp. 159–173, 2019.

[9] N. Audebert, B. Le Saux, and S. Lefèvre, "Generative adversarial networks for realistic synthesis of hyperspectral samples," in *Proc. IEEE IGARSS*, 2018, pp. 4359–4362.

[10] J. Bai, R. Feng, L. Wang, Y. Zhong, and L. Zhang, "Weakly supervised convolutional neural networks for hyperspectral unmixing," in *2021 IEEE International Geoscience and Remote Sensing Symposium IGARSS*, IEEE, 2021, pp. 3857–3860.

[11] J. Bai, R. Feng, L. Wang, Y. Zhong, and L. Zhang, "Weakly supervised convolutional neural networks for hyperspectral unmixing," in *Proc. IEEE IGARSS*, 2021, pp. 3857–3860.

[12] K. Banerjee, R. R. Gupta, K. Vyas, B. Mishra, *et al.*, "Exploring alternatives to softmax function," *arXiv preprint arXiv:2011.11538*, 2020.

[13] R. Banner, Y. Nahshan, and D. Soudry, "Post training 4-bit quantization of convolutional networks for rapid-deployment," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[14] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE transactions on neural networks*, vol. 5, no. 2, pp. 157–166, 1994.

[15] R. v. d. Berg, T. N. Kipf, and M. Welling, "Graph convolutional matrix completion," *arXiv preprint arXiv:1706.02263*, 2017.

[16] S. M. Bhakthan and A. Loganathan, "A hyperspectral unmixing model using convolutional vision transformer," *Earth Science Informatics*, vol. 17, no. 3, pp. 2255–2273, 2024.

[17] J. S. Bhatt and M. V. Joshi, "Deep learning in hyperspectral unmixing: A review," in *Igarss 2020-2020 IEEE international geoscience and remote sensing symposium*, IEEE, 2020, pp. 2189–2192.

[18] U. A. Bhatti, H. Tang, G. Wu, S. Marjan, and A. Hussain, "Deep learning with graph convolutional networks: An overview and latest applications in computational intelligence," *International Journal of Intelligent Systems*, vol. 2023, no. 1, p. 8 342 104, 2023.

[19] J. M. Bioucas-Dias, A. Plaza, G. Camps-Valls, P. Scheunders, N. Nasrabadi, and J. Chanussot, "Hyperspectral remote sensing data analysis and future challenges," *IEEE Geoscience and remote sensing magazine*, vol. 1, no. 2, pp. 6–36, 2013.

[20] J. M. Bioucas-Dias, A. Plaza, N. Dobigeon, M. Parente, Q. Du, P. Gader, and J. Chanussot, "Hyperspectral unmixing overview: Geometrical, statistical, and sparse regression-based approaches," *IEEE journal of selected topics in applied earth observations and remote sensing*, vol. 5, no. 2, pp. 354–379, 2012.

[21] J. W. Boardman, F. A. Kruse, and R. O. Green, "Mapping target signatures via partial unmixing of aviris data," in *Summaries of the Fifth Annual JPL Airborne Earth Science Workshop. Volume 1: AVIRIS Workshop*, Jet Propulsion Laboratory, 1995.

[22] L. Borawar and R. Kaur, "Resnet: Solving vanishing gradient in deep networks," in *Proceedings of International Conference on Recent Trends in Computing: ICRTC 2022*, Springer, 2023, pp. 235–247.

[23] R. A. Borsoi, T. Imbiriba, J. C. M. Bermudez, C. Richard, J. Chanussot, L. Drumetz, J.-Y. Tourneret, A. Zare, and C. Jutten, "Spectral variability in hyperspectral data unmixing: A comprehensive review," *IEEE geoscience and remote sensing magazine*, vol. 9, no. 4, pp. 223–270, 2021.

[24] A. Botev, G. Lever, and D. Barber, "Nesterov's accelerated gradient and momentum as approximations to regularised update descent," in *2017 International Joint Conference on Neural Networks (IJCNN)*, IEEE, 2017, pp. 1899–1903.

[25] T. Cai, S. Luo, K. Xu, D. He, T.-y. Liu, and L. Wang, "Graphnorm: A principled approach to accelerating graph neural network training," in *International Conference on Machine Learning*, PMLR, 2021, pp. 1204–1215.

[26] Y. Cai, J. Xie, S. Lang, J. Yang, and D. Liu, "Hyperspectral image classification using multi-branch-multi-scale residual fusion network," *Journal of Applied Remote Sensing*, vol. 15, no. 2, pp. 024 512–024 512, 2021.

[27] C. Cangea, P. Veličković, N. Jovanović, T. Kipf, and P. Liò, "Towards sparse hierarchical graph classifiers," *arXiv preprint arXiv:1811.01287*, 2018.

[28] G. Chandrashekar and F. Sahin, "A survey on feature selection methods," *Computers & electrical engineering*, vol. 40, no. 1, pp. 16–28, 2014.

[29] C.-I. Chang, "Spectral information divergence for hyperspectral image analysis," in *IEEE 1999 International Geoscience and Remote Sensing Symposium. IGARSS'99 (Cat. No. 99CH36293)*, IEEE, vol. 1, 1999, pp. 509–511.

[30] F. Chen, K. Wang, T. Van de Voorde, and T. F. Tang, "Mapping urban land cover from high spatial resolution hyperspectral data: An approach based on simultaneously unmixing similar pixels with jointly sparse spectral mixture analysis," *Remote Sensing of Environment*, vol. 196, pp. 324–342, 2017.

[31] H. Chen, T. Chen, Y. Zhang, B. Du, and A. Plaza, "Dsfc-ae: A new hyperspectral unmixing method based on deep shared fully connected autoencoder," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 2024.

[32] H. Chen, C. Wu, B. Du, L. Zhang, and L. Wang, "Change detection in multisource vhr images via deep siamese convolutional multiple-layers recurrent neural network," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 58, no. 4, pp. 2848–2864, 2019.

[33] J. Chen, C. Richard, A. Ferrari, and P. Honeine, "Nonlinear unmixing of hyperspectral data with partially linear least-squares support vector regression," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, IEEE, 2013, pp. 2174–2178.

[34] J. Chen, C. Richard, A. Ferrari, and P. Honeine, "Nonlinear unmixing of hyperspectral data with partially linear least-squares support vector regression," in *Proc. IEEE ICASSP*, 2013, pp. 2174–2178.

[35] W.-K. Chen, *Graph theory and its engineering applications*. World Scientific, 1997, vol. 5.

[36] X. Chen, "Understanding spectral graph neural network," *arXiv preprint https://arxiv.org/abs/2012.06660*, 2020.

[37] Y. Chen, N. M. Nasrabadi, and T. D. Tran, "Hyperspectral image classification via kernel sparse representation," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 51, no. 1, 217–231, Jan. 2013, ISSN: 1558-0644. DOI: 10.1109/tgrs.2012.2201730. [Online]. Available: http://dx.doi.org/10.1109/TGRS.2012.2201730.

[38] Z. Chen, L. Zhang, and L. Zhao, "Unifying spectral and spatial graph neural networks," in *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, 2024, pp. 5511–5513.

[39] M.-F. Cheng, A. Mukundan, R. Karmakar, M. A. E. Valappil, J. Jouhar, and H.-C. Wang, "Modern trends and recent applications of hyperspectral imaging: A review," *Technologies*, vol. 13, no. 5, p. 170, 2025.

[40] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[41] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[42] Y. Choukroun, E. Kravchik, F. Yang, and P. Kisilev, "Low-bit quantization of neural networks for efficient inference," in *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, IEEE, 2019, pp. 3009–3018.

[43] F. R. Chung, *Spectral graph theory*. American Mathematical Soc., 1997, vol. 92.

[44] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (elus)," *arXiv preprint https://arxiv.org/abs/1511.07289*, 2015.

[45] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[46] D. Datta, P. K. Mallick, A. K. Bhoi, M. F. Ijaz, J. Shafi, and J. Choi, "Hyperspectral image classification: Potentials, challenges, and future directions," *Computational intelligence and neuroscience*, vol. 2022, no. 1, p. 3 854 635, 2022.

[47] L. Datta, "A survey on activation functions and their relation with xavier and he normal initialization," *arXiv preprint arXiv:2004.06632*, 2020.

[48] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," *Advances in neural information processing systems*, vol. 29, 2016.

[49] H. Ding, Z. Wei, and Y. Ye, "Large-scale spectral graph neural networks via laplacian sparsification: Technical report," *arXiv preprint arXiv:2501.04570*, 2025.

[50] M. Ding, X. Fu, and X.-L. Zhao, "Fast and structured block-term tensor decomposition for hyperspectral unmixing," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 16, pp. 1691–1709, 2023.

[51] L. Drumetz, M.-A. Veganzones, S. Henrot, R. Phlypo, J. Chanussot, and C. Jutten, "Blind hyperspectral unmixing using an extended linear mixing model to address spectral variability," *IEEE Transactions on Image Processing*, vol. 25, no. 8, 3890–3905, Aug. 2016, ISSN: 1941-0042. DOI: 10.1109/tip.2016.2579259. [Online]. Available: http://dx.doi.org/10.1109/TIP.2016.2579259.

[52] Y. Dua, V. Kumar, and R. S. Singh, "Comprehensive review of hyperspectral image compression algorithms," *Optical Engineering*, vol. 59, no. 9, pp. 090 902–090 902, 2020.

[53] Y. Duan, X. Xu, T. Li, B. Pan, and Z. Shi, "Undat: Double-aware transformer for hyperspectral unmixing," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 61, pp. 1–12, 2023.

[54] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization.," *Journal of machine learning research*, vol. 12, no. 7, 2011.

[55] D. Dueck, *Affinity propagation: clustering data by passing messages*. University of Toronto Toronto, ON, Canada, 2009.

[56] R. Dwivedi, D. Dave, H. Naik, S. Singhal, R. Omer, P. Patel, B. Qian, Z. Wen, T. Shah, G. Morgan, and R. Ranjan, "Explainable ai (xai): Core ideas, techniques, and solutions," *ACM Computing Surveys*, vol. 55, no. 9, 1–33, Jan. 2023, ISSN: 1557-7341. DOI: 10.1145/3561048. [Online]. Available: http://dx.doi.org/10.1145/3561048.

[57] G. Edelman, E Gaston, T. Van Leeuwen, P. Cullen, and M. Aalders, "Hyperspectral imaging for non-contact analysis of forensic traces," *Forensic science international*, vol. 223, no. 1-3, pp. 28–39, 2012.

[58] F. Errica, M. Podda, D. Bacciu, and A. Micheli, "A fair comparison of graph neural networks for graph classification," *arXiv preprint arXiv:1912.09893*, 2019.

[59] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "Density-based spatial clustering of applications with noise," in *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD'96)*, AAAI Press, vol. 240, 1996, pp. 226–231.

[60] T. Fang, F. Zhu, and J. Chen, "Hyperspecral unmixing based on multilinear mixing model using convolutional autoencoders," *IEEE Transactions on Geoscience and Remote Sensing*, 2024.

[61] A. Feng, C. You, S. Wang, and L. Tassiulas, "Kergnns: Interpretable graph neural networks with graph kernels," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 6, 6614–6622, Jun. 2022, ISSN: 2159-5399. DOI: 10.1609/aaai.v36i6.20615. [Online]. Available: http://dx.doi.org/10.1609/aaai.v36i6.20615.

[62] F. Feng, R. Cong, S. Wei, Y. Zhang, J. Li, S. Kwong, and W. Zhang, "Unleashing correlation and continuity for hyperspectral reconstruction from rgb images," *arXiv preprint arXiv:2501.01481*, 2025.

[63] J. Feng, Y. Chen, F. Li, A. Sarkar, and M. Zhang, "How powerful are k-hop message passing graph neural networks," *Advances in Neural Information Processing Systems*, vol. 35, pp. 4776–4790, 2022.

[64] J. Feng, X. Wu, R. Shang, C. Sui, J. Li, L. Jiao, and X. Zhang, "Attention multibranch convolutional neural network for hyperspectral image classification based on adaptive region search," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 59, no. 6, pp. 5054–5070, 2020.

[65] W. Feng, J. Zhang, Y. Dong, Y. Han, H. Luan, Q. Xu, Q. Yang, E. Kharlamov, and J. Tang, "Graph random neural networks for semi-supervised learning on graphs," *Advances in neural information processing systems*, vol. 33, pp. 22 092–22 103, 2020.

[66] B. J. Frey and D. Dueck, "Clustering by passing messages between data points," *Science*, vol. 315, no. 5814, pp. 972–976, 2007. DOI: 10.1126/science.1136800. eprint: https://www.science.org/doi/pdf/10.1126/science.1136800. [Online]. Available: https://www.science.org/doi/abs/10.1126/science.1136800.

[67] W. Gacek, L. Tulczyjew, A. M. Wijata, N. Longépé, B. Le Saux, and J. Nalepa, "Estimating soil parameters from hyperspectral imagesusing ensembles of classic and deep machine learning models," in *IGARSS 2024-2024 IEEE International Geoscience and Remote Sensing Symposium*, IEEE, 2024, pp. 412–416.

[68] H. Gao, Y. Yang, S. Lei, C. Li, H. Zhou, and X. Qu, "Multi-branch fusion network for hyperspectral image classification," *Knowledge-Based Systems*, vol. 167, pp. 11–25, 2019.

[69] Q. Gao, S. Lim, and X. Jia, "Hyperspectral image classification using convolutional neural networks and multiple feature learning," *Remote Sensing*, vol. 10, no. 2, p. 299, 2018.

[70] Q. Gao, S. Lim, and X. Jia, "Hyperspectral image classification using convolutional neural networks and multiple feature learning," *Remote Sensing*, vol. 10, no. 2, p. 299, 2018.

[71] Q. Gao, S. Lim, and X. Jia, "Hyperspectral image classification using convolutional neural networks and multiple feature learning," *Remote Sensing*, vol. 10, no. 2, p. 299, 2018, ISSN: 2072-4292.

[72] Y. Gao, B. Pan, X. Xu, X. Song, and Z. Shi, "A reversible generative network for hyperspectral unmixing with spectral variability," *IEEE Transactions on Geoscience and Remote Sensing*, 2024.

[73] Y. Ge, L. Han, M. E. Paoletti, J. M. Haut, and G. Qu, "Transformer-enhanced cnn based on intensive feature for hyperspectral unmixing," *IEEE Geoscience and Remote Sensing Letters*, 2024.

[74] D. Georgiev, Á. Fernández-Galiana, S. Vilms Pedersen, G. Papadopoulos, R. Xie, M. M. Stevens, and M. Barahona, "Hyperspectral unmixing for raman spectroscopy via physics-constrained autoencoders," *Proceedings of the National Academy of Sciences*, vol. 121, no. 45, e2407439121, 2024.

[75] C. Gernigon, "Neural network quantization methods for fpga on-board processing of satellite images," Ph.D. dissertation, Université de Rennes, 2025.

[76] H. Gholamalinezhad and H. Khosravi, "Pooling methods in deep neural networks, a review," *arXiv preprint arXiv:2009.07485*, 2020.

[77] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Message passing neural networks," in *Machine learning meets quantum physics*, Springer, 2020, pp. 199–214.

[78] F. Girosi, M. Jones, and T. Poggio, "Regularization theory and neural networks architectures," *Neural computation*, vol. 7, no. 2, pp. 219–269, 1995.

[79] M. Govender, K. Chetty, and H. Bulcock, "A review of hyperspectral remote sensing and its application in vegetation and water resource studies," *Water Sa*, vol. 33, no. 2, pp. 145–151, 2007.

[80] B. Grabowski, A. M. Wijata, L. Tulczyjew, B. Le Saux, and J. Nalepa, "Soil analysis with very few labels using semi-supervised hyperspectral image classification," in *IGARSS 2024-2024 IEEE International Geoscience and Remote Sensing Symposium*, IEEE, 2024, pp. 407–411.

[81] B. Guo, S. Gunn, R. Damper, and J. Nelson, "Band selection for hyperspectral image classification using mutual information," *IEEE Geoscience and Remote Sensing Letters*, vol. 3, no. 4, pp. 522–526, 2006. DOI: 10.1109/LGRS.2006.878240.

[82] Z. Guo, C. Mu, and Y. Liu, "A multi-branch network based on weight sharing and attention mechanism for hyperspectral image classification," in *2021 IEEE International Geoscience and Remote Sensing Symposium IGARSS*, IEEE, 2021, pp. 5370–5373.

[83] S. H. Haji and A. M. Abdulazeez, "Comparison of optimization techniques based on gradient descent algorithm: A review," *PalArch's Journal of Archaeology of Egypt/Egyptology*, vol. 18, no. 4, pp. 2715–2743, 2021.

[84] X. Han, Z. Jiang, Y. Liu, J. Zhao, Q. Sun, and Y. Li, "A spatial–spectral combination method for hyperspectral band selection," *Remote Sensing*, vol. 14, no. 13, p. 3217, 2022.

[85] B. Hanin, "Which neural net architectures give rise to exploding and vanishing gradients?" *Proc. NIPS*, vol. 31, 2018.

[86] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant, "Array programming with NumPy," *Nature*, vol. 585, no. 7825, pp. 357–362, Sep. 2020. DOI: 10.1038/s41586-020-2649-2. [Online]. Available: https://doi.org/10.1038/s41586-020-2649-2.

[87] 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.

[88] D. Heinz, C. Chang, and M. Althouse, "Fully constrained least-squares based linear unmixing," in *Proc. IEEE IGARSS*, 1999, pp. 1401–1403.

[89] R. Heylen, M. Parente, and P. Gader, "A review of nonlinear hyperspectral unmixing methods," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 7, no. 6, pp. 1844–1868, 2014.

[90] S. Hochreiter, "The vanishing gradient problem during learning recurrent neural nets and problem solutions," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 6, no. 02, pp. 107–116, 1998.

[91] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[92] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[93] D. Hong, N. Yokoya, J. Chanussot, and X. X. Zhu, "An augmented linear mixing model to address spectral variability for hyperspectral unmixing," *IEEE Transactions on Image Processing*, vol. 28, no. 4, pp. 1923–1938, 2018.

[94] S. Hu and H. Li, "Hyperspectral unmixing with multi-scale convolution attention network," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 2023.

[95] 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.

[96] Y. Hu, A. Huber, J. Anumula, and S.-C. Liu, "Overcoming the vanishing gradient problem in plain recurrent networks," *arXiv preprint arXiv:1801.06105*, 2018.

[97] Q. Huang, M. Yamada, Y. Tian, D. Singh, and Y. Chang, "Graphlime: Local interpretable model explanations for graph neural networks," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 7, pp. 6968–6972, 2022.

[98] X. Huang and S. Belongie, "Arbitrary style transfer in real-time with adaptive instance normalization," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 1501–1510.

[99] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*, PMLR, 2015, pp. 448–456.

[100] M.-D. Iordache, J. M. Bioucas-Dias, and A. Plaza, "Sparse unmixing of hyperspectral data," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 49, no. 6, pp. 2014–2039, 2011.

[101] B. Jacob, S. Kligys, B. Chen, M. Zhu, M. Tang, A. Howard, H. Adam, and D. Kalenichenko, "Quantization and training of neural networks for efficient integer-arithmetic-only inference," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 2704–2713.

[102] A. Jacquet, G. Infantes, N. Meuleau, E. Benazera, S. Roussel, V. Baudoui, and J. Guerra, "Earth observation satellite scheduling with graph neural networks," *arXiv preprint arXiv:2408.15041*, 2024.

[103] T. Jastrzab, M. Myller, L. Tulczyjew, M. Blocho, M. Kawulok, A. Czornik, and J. Nalepa, "Standardized validation of vehicle routing algorithms," *Applied Intelligence*, vol. 54, no. 2, pp. 1335–1364, 2024.

[104] T. Jastrzab, M. Myller, L. Tulczyjew, M. Blocho, W. Ryczko, M. Kawulok, and J. Nalepa, "Particle swarm optimization configures the route minimization algorithm," in *International Conference on Computational Science*, Springer, 2022, pp. 80–87.

[105] S. Ji, S. Pan, E. Cambria, P. Marttinen, and S. Y. Philip, "A survey on knowledge graphs: Representation, acquisition, and applications," *IEEE transactions on neural networks and learning systems*, vol. 33, no. 2, pp. 494–514, 2021.

[106] D. Jin and B. Yang, "Graph attention convolutional autoencoder-based unsupervised nonlinear unmixing for hyperspectral images," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 2023.

[107] W. Jin, M. Qu, X. Jin, and X. Ren, "Recurrent event network: Autoregressive structure inference over temporal knowledge graphs," *arXiv preprint https://arxiv.org/abs/1904.05530*, 2019.

[108] A. Jović, K. Brkić, and N. Bogunović, "A review of feature selection methods with applications," in *2015 38th international convention on information and communication technology, electronics and microelectronics (MIPRO)*, Ieee, 2015, pp. 1200–1205.

[109] S. Kapoor and A. Narayanan, "Leakage and the reproducibility crisis in machine-learning-based science," *Patterns*, vol. 4, no. 9, 2023.

[110] J. Karlgren, M. Sahlgren, F. Olsson, F. Espinoza, and O. Hamfors, "Usefulness of sentiment analysis," in *European Conference on Information Retrieval*, Springer, 2012, pp. 426–435.

[111] D. Kavran, D. Mongus, B. Žalik, and N. Lukač, "Graph neural network-based method of spatiotemporal land cover mapping using satellite imagery," *Sensors*, vol. 23, no. 14, p. 6648, 2023.

[112] F. Khajehrayeni and H. Ghassemian, "Hyperspectral unmixing using deep convolutional autoencoders in a supervised scenario," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 13, pp. 567–576, 2020.

[113] F. Khajehrayeni and H. Ghassemian, "Hyperspectral unmixing using deep convolutional autoencoders in a supervised scenario," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 13, pp. 567–576, 2020.

[114] F. Khajehrayeni and H. Ghassemian, "Hyperspectral unmixing using deep convolutional autoencoders in a supervised scenario," *IEEE J-STARS*, vol. 13, pp. 567–576, 2020. DOI: 10.1109/JSTARS.2020.2966512.

[115] B.-H. Kim and J. C. Ye, "Understanding graph isomorphism network for rs-fmri functional connectivity analysis," *Frontiers in neuroscience*, vol. 14, p. 630, 2020.

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

[117] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[118] T. N. Kipf and M. Welling, "Variational graph auto-encoders," *arXiv preprint arXiv:1611.07308*, 2016.

[119] B. Knyazev, G. W. Taylor, and M. Amer, "Understanding attention and generalization in graph neural networks," *Advances in neural information processing systems*, vol. 32, 2019.

[120] M. M. Komarnicki, M. W. Przewozniczek, H. Kwasnicka, and K. Walkowiak, "Incremental recursive ranking grouping for large-scale global optimization," *IEEE Transactions on Evolutionary Computation*, vol. 27, no. 5, pp. 1498–1513, 2023. DOI: 10.1109/TEVC.2022.3216968.

[121] F. Kruse, "Identification and mapping of minerals in drill core using hyperspectral image analysis of infrared reflectance spectra," *International journal of remote sensing*, vol. 17, no. 9, pp. 1623–1632, 1996.

[122] N. Kumar, P. Uppala, K. Duddu, H. Sreedhar, V. Varma, G. Guzman, M. Walsh, and A. Sethi, "Hyperspectral tissue image segmentation using semi-supervised nmf and hierarchical clustering," *IEEE Transactions on Medical Imaging*, vol. 38, no. 5, 1304–1313, May 2019, ISSN: 1558-254X. DOI: 10.1109/tmi.2018.2883301. [Online]. Available: http://dx.doi.org/10.1109/TMI.2018.2883301.

[123] C. Kwan, B. Ayhan, B. Budavari, Y. Lu, D. Perez, J. Li, S. Bernabe, and A. Plaza, "Deep learning for land cover classification using only a few bands," *Remote Sensing*, vol. 12, no. 12, p. 2000, 2020.

[124] D.-Y. Lan, P.-J. He, Y.-P. Qi, F. Lü, and H. Zhang, "Unveiling the non-linear effects of water and oil on hyperspectral imaging-based characterization of solid waste by hyperspectral unmixing," *Waste Management*, vol. 190, pp. 251–260, 2024.

[125] D. Landgrebe, "Hyperspectral image data analysis," *IEEE Signal processing magazine*, vol. 19, no. 1, pp. 17–28, 2002.

[126] D. Laptev, N. Savinov, J. M. Buhmann, and M. Pollefeys, "Ti-pooling: Transformation-invariant pooling for feature learning in convolutional neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 289–297.

[127] H. Le, T. Tran, and S. Venkatesh, "Self-attentive associative memory," in *Proc. ICML*, vol. 119, 2020, pp. 5682–5691.

[128] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[129] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[130] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[131] J. Lee, I. Lee, and J. Kang, "Self-attention graph pooling," in *International conference on machine learning*, pmlr, 2019, pp. 3734–3743.

[132] J. Lei Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *ArXiv e-prints*, arXiv–1607, 2016.

[133] C. Li, R. Cai, and J. Yu, "An attention-based 3d convolutional autoencoder for few-shot hyperspectral unmixing and classification," *Remote Sensing*, vol. 15, no. 2, p. 451, 2023.

[134] H. Li, D. Li, M. Gong, J. Li, A. K. Qin, L. Xing, and F. Xie, "Sparse hyperspectral unmixing with preference-based evolutionary multiobjective multitasking optimization," *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2024.

[135] J. Li, J. M. Bioucas-Dias, and A. Plaza, "Semisupervised hyperspectral image segmentation using multinomial logistic regression with active learning," *IEEE Transactions on Geoscience and Remote Sensing*, Nov. 2010, ISSN: 1558-0644. DOI: 10.1109/tgrs.2010.2060550. [Online]. Available: http://dx.doi.org/10.1109/TGRS.2010.2060550.

[136] S. Li, W. Song, L. Fang, Y. Chen, P. Ghamisi, and J. A. Benediktsson, "Deep learning for hyperspectral image classification: An overview," *IEEE transactions on geoscience and remote sensing*, vol. 57, no. 9, pp. 6690–6709, 2019.

[137] W. Li, G. Wu, F. Zhang, and Q. Du, "Hyperspectral image classification using deep pixel-pair features," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 2, pp. 844–853, 2016.

[138] X. Li, Y. Zhou, N. Dvornek, M. Zhang, S. Gao, J. Zhuang, D. Scheinost, L. H. Staib, P. Ventola, and J. S. Duncan, "Braingnn: Interpretable brain graph neural network for fmri analysis," *Medical Image Analysis*, vol. 74, p. 102 233, Dec. 2021, ISSN: 1361-8415. DOI: 10.1016/j.media.2021.102233. [Online]. Available: http://dx.doi.org/10.1016/j.media.2021.102233.

[139] Y. Li, X. Zhang, and D. Chen, "Csrnet: Dilated convolutional neural networks for understanding the highly congested scenes," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 1091–1100.

[140] Y. Li, X. Zhang, and D. Chen, "Csrnet: Dilated convolutional neural networks for understanding the highly congested scenes," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 1091–1100.

[141] G. A. Licciardi and F. Del Frate, "Pixel unmixing in hyperspectral data by means of neural networks," *IEEE transactions on Geoscience and remote sensing*, vol. 49, no. 11, pp. 4163–4172, 2011.

[142] G. A. Licciardi and F. Del Frate, "Pixel unmixing in hyperspectral data by means of neural networks," *IEEE transactions on Geoscience and remote sensing*, vol. 49, no. 11, pp. 4163–4172, 2011.

[143] C. Liu, Y. Zhan, J. Wu, C. Li, B. Du, W. Hu, T. Liu, and D. Tao, "Graph pooling for graph neural networks: Progress, challenges, and opportunities," *arXiv preprint arXiv:2204.07321*, 2022.

[144] J. Liu, G. P. Ong, and X. Chen, "Graphsage-based traffic speed forecasting for segment network with sparse data," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 3, pp. 1755–1766, 2020.

[145] M. Liu, H. Gao, and S. Ji, "Towards deeper graph neural networks," in *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, 2020, pp. 338–348.

[146] W. Liu, Y. Zhang, J. Wang, Y. He, J. Caverlee, P. P. Chan, D. S. Yeung, and P.-A. Heng, "Item relationship graph neural networks for e-commerce," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 9, pp. 4785–4799, 2021.

[147] X. Liu, J. Ding, W. Jin, H. Xu, Y. Ma, Z. Liu, and J. Tang, "Graph neural networks with adaptive residual," *Advances in Neural Information Processing Systems*, vol. 34, pp. 9720–9733, 2021.

[148] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, Oct. 2021, pp. 10 012–10 022.

[149] Z. Liu, Y. Wang, K. Han, W. Zhang, S. Ma, and W. Gao, "Post-training quantization for vision transformer," *Advances in Neural Information Processing Systems*, vol. 34, pp. 28 092–28 103, 2021.

[150] P. R. Lorenzo, L. Tulczyjew, M. Marcinkiewicz, and J. Nalepa, "Hyperspectral band selection using attention-based convolutional neural networks," *IEEE Access*, vol. 8, pp. 42 384–42 403, 2020.

[151] P. R. Lorenzo, L. Tulczyjew, M. Marcinkiewicz, and J. Nalepa, "Hyperspectral band selection using attention-based convolutional neural networks," *IEEE Access*, vol. 8, pp. 42 384–42 403, 2020.

[152] B. Lu, P. D. Dao, J. Liu, Y. He, and J. Shang, "Recent advances of hyperspectral imaging technology and applications in agriculture," *Remote Sensing*, vol. 12, no. 16, p. 2659, 2020.

[153] G. Lu and B. Fei, "Medical hyperspectral imaging: A review," *Journal of biomedical optics*, vol. 19, no. 1, p. 010 901, 2014.

[154] J. Lu, H. Liu, Y. Yao, S. Tao, Z. Tang, and J. Lu, "Hsi road: A hyper spectral image dataset for road segmentation," in *2020 IEEE International Conference on Multimedia and Expo (ICME)*, IEEE, 2020, pp. 1–6.

[155] L. Lu, Y. Shin, Y. Su, and G. E. Karniadakis, "Dying relu and initialization: Theory and numerical examples," *arXiv preprint arXiv:1903.06733*, 2019.

[156] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," *NeurIPS*, 2017.

[157] C. Ma, L. Ma, Y. Zhang, J. Sun, X. Liu, and M. Coates, "Memory augmented graph neural networks for sequential recommendation," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, 2020, pp. 5045–5052.

[158] Z. Ma, J. Xuan, Y. G. Wang, M. Li, and P. Liò, "Path integral based convolution and pooling for graph neural networks," *Advances in Neural Information Processing Systems*, vol. 33, pp. 16 421–16 433, 2020.

[159] A. Mao, M. Mohri, and Y. Zhong, "Cross-entropy loss functions: Theoretical analysis and applications," in *International conference on Machine learning*, PMLR, 2023, pp. 23 803–23 828.

[160] V. Md, S. Misra, G. Ma, R. Mohanty, E. Georganas, A. Heinecke, D. Kalamkar, N. K. Ahmed, and S. Avancha, "Distgnn: Scalable distributed training for large-scale graph neural networks," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, 2021, pp. 1–14.

[161] S. A. Medjahed and M. Ouali, "A hybrid approach for supervised spectral band selection in hyperspectral images classification," *Computación y Sistemas*, vol. 24, no. 1, pp. 213–219, 2020.

[162] T. Mikolov, A. Deoras, D. Povey, L. Burget, and J. Černockỳ, "Strategies for training large scale neural network language models," in *2011 IEEE Workshop on Automatic Speech Recognition & Understanding*, IEEE, 2011, pp. 196–201.

[163] T. Mikolov, M. Karafiát, L. Burget, J. Černockỳ, and S. Khudanpur, "Recurrent neural network based language model," in *Eleventh annual conference of the international speech communication association*, 2010.

[164] T. Mikolov, S. Kombrink, L. Burget, J. Černockỳ, and S. Khudanpur, "Extensions of recurrent neural network language model," in *2011 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, IEEE, 2011, pp. 5528–5531.

[165] 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.

[166] S. A. Myers, A. Sharma, P. Gupta, and J. Lin, "Information network or social network? the structure of the twitter follow graph," in *Proceedings of the 23rd international conference on world wide web*, 2014, pp. 493–498.

[167] M. Nagel, R. A. Amjad, M. Van Baalen, C. Louizos, and T. Blankevoort, "Up or down? adaptive rounding for post-training quantization," in *International conference on machine learning*, PMLR, 2020, pp. 7197–7206.

[168] M. Nagel, M. v. Baalen, T. Blankevoort, and M. Welling, "Data-free quantization through weight equalization and bias correction," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 1325–1334.

[169] M. Nagel, M. Fournarakis, Y. Bondarenko, and T. Blankevoort, "Overcoming oscillations in quantization-aware training," in *International Conference on Machine Learning*, PMLR, 2022, pp. 16 318–16 330.

[170] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Icml*, 2010.

[171] J. Nalepa, B. Le Saux, N. Longépé, L. Tulczyjew, M. Myller, M. Kawulok, K. Smykala, and M. Gumiela, "The hyperview challenge: Estimating soil parameters from hyperspectral images," in *2022 IEEE International Conference on Image Processing (ICIP)*, IEEE, 2022, pp. 4268–4272.

[172] J. Nalepa, M. Myller, M. Cwiek, L. Zak, T. Lakota, L. Tulczyjew, and M. Kawulok, "Towards on-board hyperspectral satellite image segmentation: Understanding robustness of deep learning through simulating acquisition conditions," *Remote Sensing*, vol. 13, no. 8, p. 1532, Apr. 2021, ISSN: 2072-4292. DOI: 10.3390/rs13081532. [Online]. Available: http://dx.doi.org/10.3390/rs13081532.

[173] J. Nalepa, M. Myller, M. Cwiek, L. Zak, T. Lakota, L. Tulczyjew, and M. Kawulok, "Towards on-board hyperspectral satellite image segmentation: Understanding robustness of deep learning through simulating acquisition conditions," *Remote Sensing*, vol. 13, no. 8, p. 1532, 2021.

[174] J. Nalepa, M. Myller, M. Cwiek, L. Zak, T. Lakota, L. Tulczyjew, and M. Kawulok, "Towards on-board hyperspectral satellite image segmentation: Understanding robustness of deep learning through simulating acquisition conditions," *Remote Sensing*, vol. 13, no. 8, 2021, ISSN: 2072-4292.

[175] J. Nalepa, M. Myller, Y. Imai, K.-I. Honda, T. Takeda, and M. Antoniak, "Unsupervised segmentation of hyperspectral images using 3-d convolutional autoencoders," *IEEE Geoscience and Remote Sensing Letters*, vol. 17, no. 11, pp. 1948–1952, 2020.

[176] J. Nalepa, M. Myller, and M. Kawulok, "Validating hyperspectral image segmentation," *IEEE Geoscience and Remote Sensing Letters*, vol. 16, no. 8, pp. 1264–1268, 2019.

[177] J. Nalepa, M. Myller, and M. Kawulok, "Validating hyperspectral image segmentation," *IEEE GRSL*, vol. 16, no. 8, pp. 1264–1268, 2019, ISSN: 1558-0571.

[178] J. Nalepa, M. Myller, L. Tulczyjew, and M. Kawulok, "Deep ensembles for hyperspectral image data classification and unmixing," *Remote Sensing*, vol. 13, no. 20, p. 4133, 2021.

[179] J. Nalepa, L. Tulczyjew, B. Le Saux, N. Longépé, B. Ruszczak, A. M. Wijata, K. Smykala, M. Myller, M. Kawulok, R. S. Kuzu, *et al.*, "Estimating soil parameters from hyperspectral images: A benchmark dataset and the outcome of the hyperview challenge," *IEEE Geoscience and Remote Sensing Magazine*, 2024.

[180] J. Nalepa, L. Tulczyjew, M. Myller, and M. Kawulok, "Hyperspectral image classification using spectral-spatial convolutional neural networks," in *IGARSS 2020-2020 IEEE International Geoscience and Remote Sensing Symposium*, IEEE, 2020, pp. 866–869.

[181] J. Nalepa, L. Tulczyjew, M. Myller, and M. Kawulok, "Hyperspectral image classification using spectral-spatial convolutional neural networks," in *IGARSS 2020-2020 IEEE International Geoscience and Remote Sensing Symposium*, IEEE, 2020, pp. 866–869.

[182] J. Nalepa, L. Tulczyjew, M. Myller, and M. Kawulok, "Hyperspectral image classification using spectral-spatial convolutional neural networks," in *Proc. IGARSS*, 2020, pp. 866–869.

[183] J. Nascimento and J. Dias, "Vertex component analysis: A fast algorithm to unmix hyperspectral data," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 43, no. 4, 898–910, Apr. 2005, ISSN: 0196-2892. DOI: 10.1109/tgrs.2005.844293. [Online]. Available: http://dx.doi.org/10.1109/TGRS.2005.844293.

[184] A. Y. Ng, "Feature selection, l 1 vs. l 2 regularization, and rotational invariance," in *Proceedings of the twenty-first international conference on Machine learning*, 2004, p. 78.

[185] I. Nusrat and S.-B. Jang, "A comparison of regularization techniques in deep neural networks," *Symmetry*, vol. 10, no. 11, p. 648, 2018.

[186] C. Nwankpa, W. Ijomah, A. Gachagan, and S. Marshall, "Activation functions: Comparison of trends in practice and research for deep learning," *arXiv preprint arXiv:1811.03378*, 2018.

[187] S. K. Pal and S. Mitra, "Multilayer perceptron, fuzzy sets, classifiaction," 1992.

[188] B. Palsson, J. Sigurdsson, J. R. Sveinsson, and M. O. Ulfarsson, "Hyperspectral unmixing using a neural network autoencoder," *IEEE Access*, vol. 6, pp. 25 646–25 656, 2018.

[189] B. Palsson, J. R. Sveinsson, and M. O. Ulfarsson, "Spectral-spatial hyperspectral unmixing using multitask learning," *IEEE Access*, vol. 7, pp. 148 861–148 872, 2019.

[190] E. Park, S. Yoo, and P. Vajda, "Value-aware quantization for training and inference of neural networks," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 580–595.

[191] L. Parra, C. Spence, P. Sajda, A. Ziehe, and K.-R. Müller, "Unmixing hyperspectral data," *Advances in neural information processing systems*, vol. 12, 1999.

[192] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," in *International conference on machine learning*, PMLR, 2013, pp. 1310–1318.

[193] R. N. Patro, S. Subudhi, P. K. Biswal, F. Dell'acqua, and B. Biswal, "A hyperspectral band selection strategy based on adjacency clustering and local structural correlation," *International Journal of Remote Sensing*, vol. 45, no. 3, pp. 848–862, 2024.

[194] H. Petersson, D. Gustafsson, and D. Bergstrom, "Hyperspectral image analysis using deep learning—a review," in *2016 sixth international conference on image processing theory, tools and applications (IPTA)*, IEEE, 2016, pp. 1–6.

[195] M. W. Przewozniczek and M. M. Komarnicki, "Empirical problem decomposition — the key to the evolutionary effectiveness in solving a large-scale non-binary discrete real-world problem," *Applied Soft Computing*, vol. 113, p. 107 864, 2021, ISSN: 1568-4946. DOI: https://doi.org/10.1016/j.asoc.2021.107864. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1568494621007869.

[196] M. W. Przewozniczek, M. M. Komarnicki, and B. Frej, "Direct linkage discovery with empirical linkage learning," in *Proceedings of the Genetic and Evolutionary Computation Conference*, ser. GECCO '21, Association for Computing Machinery, 2021, 609–617, ISBN: 9781450383509. DOI: 10.1145/3449639.3459333. [Online]. Available: https://doi.org/10.1145/3449639.3459333.

[197] K. Qu and Z. Li, "A fast sparse nmf optimization algorithm for hyperspectral unmixing," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 2023.

[198] K. Qu, Z. Li, C. Wang, F. Luo, and W. Bao, "Hyperspectral unmixing using higher-order graph regularized nmf with adaptive feature selection," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 61, pp. 1–15, 2023.

[199] E. Ranjan, S. Sanyal, and P. Talukdar, "Asap: Adaptive structure aware pooling for learning hierarchical graph representations," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, 2020, pp. 5470–5477.

[200] A. D. Rasamoelina, F. Adjailia, and P. Sinčák, "A review of activation function for artificial neural network," in *2020 IEEE 18th World Symposium on Applied Machine Intelligence and Informatics (SAMI)*, IEEE, 2020, pp. 281–286.

[201] B. Rasti, B. Koirala, P. Scheunders, and P. Ghamisi, "UnDIP: Hyperspectral unmixing using deep image prior," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 60, pp. 1–15, 2021.

[202] P. Reiser, M. Neubert, A. Eberhard, L. Torresi, C. Zhou, C. Shao, H. Metni, C. van Hoesel, H. Schopmans, T. Sommer, *et al.*, "Graph neural networks for materials science and chemistry," *Communications Materials*, vol. 3, no. 1, p. 93, 2022.

[203] B. Remeseiro and V. Bolon-Canedo, "A review of feature selection methods in medical applications," *Computers in biology and medicine*, vol. 112, p. 103 375, 2019.

[204] M. T. Ribeiro, S. Singh, and C. Guestrin, "Why should i trust you?: Explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD*, 2016.

[205] C. Rotaru, T. Graf, and J. Zhang, "Color image segmentation in hsi space for automotive applications," *Journal of Real-Time Image Processing*, vol. 3, pp. 311–322, 2008.

[206] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv preprint arXiv:1609.04747*, 2016.

[207] G. Salha, R. Hennequin, and M. Vazirgiannis, "Keep it simple: Graph autoencoders without graph convolutional networks," *arXiv preprint arXiv:1910.00942*, 2019.

[208] A. Santara, K. Mani, P. Hatwar, A. Singh, A. Garg, K. Padia, and P. Mitra, "Bass net: Band-adaptive spectral-spatial feature learning neural network for hyperspectral image classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 9, pp. 5293–5301, 2017.

[209] D Santos, J Cardoso-Fernandes, A Lima, and A. Teodoro, "The potential of spectral unmixing method applied to prisma hyperspectral images in the identification of li minerals: An evaluation for prospecting purposes," in *Earth Resources and Environmental Remote Sensing/GIS Applications XIII*, SPIE, vol. 12268, 2022, pp. 257–271.

[210] D. Scherer, A. Müller, and S. Behnke, "Evaluation of pooling operations in convolutional architectures for object recognition," in *International conference on artificial neural networks*, Springer, 2010, pp. 92–101.

[211] R. R. Selvaraju *et al.*, "Grad-cam: Visual explanations from deep networks via gradient-based localization," in *ICCV*, 2017.

[212] A. Severance and G. G. Lemieux, "Embedded supercomputing in fpgas with the vectorblox mxp matrix processor," in *2013 International Conference on Hardware/Software Codesign and System Synthesis (CODES+ ISSS)*, IEEE, 2013, pp. 1–10.

[213] P. Shippert *et al.*, "Why use hyperspectral imagery?" *Photogrammetric engineering and remote sensing*, vol. 70, no. 4, pp. 377–396, 2004.

[214] H. Shirani-Mehr, "Applications of deep learning to sentiment analysis of movie reviews," *Technical report*, 2014.

[215] A. Signoroni, M. Savardi, A. Baronio, and S. Benini, "Deep learning meets hyperspectral image analysis: A multidisciplinary review," *Journal of imaging*, vol. 5, no. 5, p. 52, 2019.

[216] T. Snelleman, B. M. Renting, H. H. Hoos, and J. N. van Rijn, "Edge-based graph component pooling," *arXiv preprint arXiv:2409.11856*, 2024.

[217] F.-X. Song and S.-W. Deng, "First-order graph trend filtering for sparse hyperspectral unmixing," *IEEE Geoscience and Remote Sensing Letters*, 2023.

[218] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[219] Y. Su, J. Li, A. Plaza, A. Marinoni, P. Gamba, and S. Chakravortty, "Daen: Deep autoencoder networks for hyperspectral unmixing," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 57, no. 7, pp. 4309–4321, 2019.

[220] Y. Su, Z. Zhu, L. Gao, A. Plaza, P. Li, X. Sun, and X. Xu, "Daan: A deep autoencoder-based augmented network for blind multilinear hyperspectral unmixing," *IEEE Transactions on Geoscience and Remote Sensing*, 2024.

[221] W. Sun and Q. Du, "Hyperspectral band selection: A review," *IEEE Geoscience and Remote Sensing Magazine*, vol. 7, no. 2, pp. 118–139, 2019.

[222] M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic attribution for deep networks," in *International Conference on Machine Learning*, 2017.

[223] A. Szalay and J. Gray, "Science in an exponential world," *Nature*, vol. 440, no. 7083, pp. 413–414, 2006.

[224] X. Tao, B. Koirala, A. Plaza, and P. Scheunders, "A new dual-feature fusion network for enhanced hyperspectral unmixing," *IEEE Transactions on Geoscience and Remote Sensing*, 2024.

[225] X. Tao, M. E. Paoletti, Z. Wu, J. M. Haut, P. Ren, and A. Plaza, "An abundance-guided attention network for hyperspectral unmixing," *IEEE Transactions on Geoscience and Remote Sensing*, 2024.

[226] T. Tarasiewicz, L. Tulczyjew, M. Myller, M. Kawulok, N. Longépé, and J. Nalepa, "Extracting high-resolution cultivated land maps from sentinel-2 image series," in *IGARSS 2022-2022 IEEE International Geoscience and Remote Sensing Symposium*, IEEE, 2022, pp. 175–178.

[227] G. Tejasree and L. Agilandeeswari, "An extensive review of hyperspectral image classification and prediction: Techniques and challenges," *Multimedia Tools and Applications*, vol. 83, no. 34, pp. 80 941–81 038, 2024.

[228] D. Thierens and P. A. N. Bosman, "Hierarchical problem solving with the linkage tree genetic algorithm," in *Proceedings of the 15th Annual Conference on Genetic and Evolutionary Computation*, ser. GECCO '13, Association for Computing Machinery, 2013, 877–884, ISBN: 9781450319638. DOI: 10.1145/2463372.2463477. [Online]. Available: https://doi.org/10.1145/2463372.2463477.

[229] R. Tinós, M. Przewozniczek, D. Whitley, and F. Chicano, "Genetic algorithm with linkage learning," in *Proceedings of the Genetic and Evolutionary Computation Conference*, ser. GECCO '23, Association for Computing Machinery, 2023, 981–989, ISBN: 9798400701191. DOI: 10.1145/3583131.3590349. [Online]. Available: https://doi.org/10.1145/3583131.3590349.

[230] R. Tinós, M. Przewozniczek, D. Whitley, and F. Chicano, "Genetic algorithm with linkage learning," in *Proceedings of the Genetic and Evolutionary Computation Conference*, 2023, pp. 981–989.

[231] L. Tulczyjew, I. Biruk, M. Bilgic, C. Abondo, and N. Weill, "Pcapvision: Pcap-based high-velocity and large-volume network failure detection," in *Proceedings of the 2024 SIGCOMM Workshop on Networks for AI Computing*, 2024, pp. 26–33.

[232] L. Tulczyjew, K. Jarrah, C. Abondo, D. Bennett, and N. Weill, "Llmcap: Large language model for unsupervised pcap failure detection," in *2024 IEEE International Conference on Communications Workshops (ICC Workshops)*, IEEE, 2024, pp. 1559–1565.

[233] L. Tulczyjew, M. Kawulok, N. Longépé, B. Le Saux, and J. Nalepa, "A multi-branch convolutional neural network for hyperspectral unmixing," *IEEE Geoscience and Remote Sensing Letters*, vol. 19, pp. 1–5, 2022.

[234] L. Tulczyjew, M. Kawulok, N. Longépé, B. Le Saux, and J. Nalepa, "Graph neural networks extract high-resolution cultivated land maps from sentinel-2 image series," *IEEE Geoscience and Remote Sensing Letters*, vol. 19, pp. 1–5, 2022.

[235] L. Tulczyjew, M. Kawulok, N. Longépé, B. Le Saux, and J. Nalepa, "Unbiased validation of hyperspectral unmixing algorithms," in *IGARSS 2023-2023 IEEE International Geoscience and Remote Sensing Symposium*, IEEE, 2023, pp. 7344–7347.

[236] L. Tulczyjew, M. Kawulok, and J. Nalepa, "Unsupervised feature learning using recurrent neural nets for segmenting hyperspectral images," *IEEE Geoscience and Remote Sensing Letters*, vol. 18, no. 12, pp. 2142–2146, 2020.

[237] L. Tulczyjew, M. Myller, M. Kawulok, D. Kostrzewa, and J. Nalepa, "Predicting risk of satellite collisions using machine learning," *Journal of Space Safety Engineering*, vol. 8, no. 4, pp. 339–344, 2021.

[238] L. Tulczyjew and J. Nalepa, "Investigating the impact of the training set size on deep learning-powered hyperspectral unmixing," in *2021 IEEE International Geoscience and Remote Sensing Symposium IGARSS*, IEEE, 2021, pp. 2024–2027.

[239] L. Tulczyjew and J. Nalepa, "Investigating the impact of the training set size on deep learning-powered hyperspectral unmixing," in *IGARSS 2021 - 2021 IEEE International Geoscience and Remote Sensing Symposium*, in press, 2021.

[240] L. Tulczyjew, M. Przewozniczek, R. Tinós, A. M. Wijata, and J. Nalepa, "Cannibal unveils the hidden gems: Hyperspectral band selection via clustering of weighted variable interaction graphs," in *Proceedings of the Genetic and Evolutionary Computation Conference*, 2024, pp. 412–421.

[241] R. Vaddi, B. P. Kumar, P. Manoharan, L Agilandeeswari, and V Sangeetha, "Strategies for dimensionality reduction in hyperspectral remote sensing: A comprehensive overview," *The Egyptian Journal of Remote Sensing and Space Sciences*, vol. 27, no. 1, pp. 82–92, 2024.

[242] B. Valentin, L. Gale, J. Lathouwers, G. Smith, M. Cerri, M. Pellegrini, F. Ferrante, J. Nalepa, L. Tulczyjew, J. Sadel, *et al.*, "Aiopen–platform extensions with ai capabilities,"

[243] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *arXiv preprint arXiv:1710.10903*, 2017.

[244] L. Wang and X. Jia, "Integration of soft and hard classifications using extended support vector machines," *IEEE Geoscience and Remote Sensing Letters*, vol. 6, no. 3, pp. 543–547, 2009.

[245] P. Wang, Q. Chen, X. He, and J. Cheng, "Towards accurate post-training network quantization via bit-split and stitching," in *International Conference on Machine Learning*, PMLR, 2020, pp. 9847–9856.

[246] P. Wang, R. Liu, and L. Zhang, "Mat-net: Multiscale aggregation transformer network for hyperspectral unmixing," *IEEE Transactions on Geoscience and Remote Sensing*, 2024.

[247] W. Wang, Y. Qian, and Y. Y. Tang, "Hypergraph-regularized sparse nmf for hyperspectral unmixing," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 9, no. 2, 681–694, Feb. 2016, ISSN: 2151-1535. DOI: 10.1109/jstars.2015.2508448. [Online]. Available: http://dx.doi.org/10.1109/JSTARS.2015.2508448.

[248] X. Wang, N. Liu, H. Han, and C. Shi, "Self-supervised heterogeneous graph neural network with co-contrastive learning," in *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, 2021, pp. 1726–1736.

[249] X. Wang and G. Wang, "Hyperspectral band selection based on adaptive neighborhood grouping and local structure correlation," *Journal of Sensors*, vol. 2021, no. 1, p. 5 530 385, 2021.

[250] Y. Wang, X. Ou, H.-J. He, and M. Kamruzzaman, "Advancements, limitations and challenges in hyperspectral imaging for comprehensive assessment of wheat quality: An up-to-date review," *Food Chemistry: X*, vol. 21, p. 101 235, 2024.

[251] J. Wei and X. Wang, "An overview on linear unmixing of hyperspectral data," *Mathematical Problems in Engineering*, vol. 2020, no. 1, p. 3 735 403, 2020.

[252] X. Wei, W. Liu, L. Chen, L. Ma, H. Chen, and Y. Zhuang, "Fpga-based hybrid-type implementation of quantized neural networks for remote sensing applications," *Sensors*, vol. 19, no. 4, p. 924, 2019.

[253] A. M. Wijata, T. Lakota, M. Cwiek, B. Ruszczak, M. Gumiela, L. Tulczyjew, A. Bartoszek, N. Longépé, K. Smykala, and J. Nalepa, "Intuition-1: Toward in-orbit bare soil detection using spectral vegetation indices," in *IGARSS 2024-2024 IEEE International Geoscience and Remote Sensing Symposium*, IEEE, 2024, pp. 1708–1712.

[254] M. E. Winter, "N-findr: An algorithm for fast autonomous spectral end-member determination in hyperspectral data," in *Imaging Spectrometry V*, M. R. Descour and S. S. Shen, Eds., SPIE, Oct. 1999. DOI: 10.1117/12.366289. [Online]. Available: http://dx.doi.org/10.1117/12.366289.

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

[256] J. Wu, C. Leng, Y. Wang, Q. Hu, and J. Cheng, "Quantized convolutional neural networks for mobile devices," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 4820–4828.

[257] Y. Wu and S. Misra, "Intelligent image segmentation for organic-rich shales using random forest, wavelet transform, and hessian matrix," *IEEE Geoscience and Remote Sensing Letters*, vol. 17, no. 7, 1144–1147, Jul. 2020, ISSN: 1558-0571. DOI: 10.1109/lgrs.2019.2943849. [Online]. Available: http://dx.doi.org/10.1109/LGRS.2019.2943849.

[258] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip, "A comprehensive survey on graph neural networks," *IEEE transactions on neural networks and learning systems*, vol. 32, no. 1, pp. 4–24, 2020.

[259] S. Xiao, S. Wang, Y. Dai, and W. Guo, "Graph neural networks in node classification: Survey and evaluation," *Machine Vision and Applications*, vol. 33, no. 1, p. 4, 2022.

[260] B. Xu, N. Wang, T. Chen, and M. Li, "Empirical evaluation of rectified activations in convolutional network," *arXiv preprint arXiv:1505.00853*, 2015.

[261] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?" *arXiv preprint arXiv:1810.00826*, 2018.

[262] M. Xu, J. Xu, S. Liu, H. Sheng, B. Shen, and K. Hou, "Stationary wavelet convolutional network with generative feature learning for hyperspectral unmixing," *IEEE Transactions on Geoscience and Remote Sensing*, 2024.

[263] M. Xu, J. Xu, S. Liu, H. Sheng, and Z. Yang, "Multi-scale convolutional mask network for hyperspectral unmixing," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 2024.

[264] T. Yang, S. Li, M. Song, C. Yu, and H. Bao, "Low-rank and sparse nmf based on compression and correlation sensing for hyperspectral unmixing," *Infrared Physics & Technology*, vol. 141, p. 105 464, 2024.

[265] Z. Yang, M. Xu, S. Liu, H. Sheng, and J. Wan, "Ust-net: A u-shaped transformer network using shifted windows for hyperspectral unmixing," *IEEE Transactions on Geoscience and Remote Sensing*, 2023.

[266] W. Yin, K. Kann, M. Yu, and H. Schütze, "Comparative study of cnn and rnn for natural language processing," *arXiv preprint arXiv:1702.01923*, 2017.

[267] Z. Ying, J. You, C. Morris, X. Ren, W. Hamilton, and J. Leskovec, "Hierarchical graph representation learning with differentiable pooling," *Advances in neural information processing systems*, vol. 31, 2018.

[268] Y. You, T. Chen, Y. Sui, T. Chen, Z. Wang, and Y. Shen, "Graph contrastive learning with augmentations," *Advances in neural information processing systems*, vol. 33, pp. 5812–5823, 2020.

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

[270] V. Zaigrajew, H. Baniecki, L. Tulczyjew, A. M. Wijata, J. Nalepa, N. Longépé, and P. Biecek, "Red teaming models for hyperspectral image analysis using explainable ai," *arXiv preprint arXiv:2403.08017*, 2024.

[271] M. D. Zeiler, "Adadelta: An adaptive learning rate method," *arXiv preprint arXiv:1212.5701*, 2012.

[272] H. Zhai, H. Zhang, L. Zhang, and P. Li, "Cloud/shadow detection based on spectral indices for multi/hyperspectral optical remote sensing imagery," *ISPRS journal of photogrammetry and remote sensing*, vol. 144, pp. 235–253, 2018.

[273] J. Zhang, T. He, S. Sra, and A. Jadbabaie, "Why gradient clipping accelerates training:
a theoretical justification for adaptivity," *arXiv preprint* https://arxiv.org/abs/1905.11881, 2019.

[274] M. Zhang, M. Gong, and Y. Chan, "Hyperspectral band selection based on multi-objective optimization with high information and low redundancy," *Applied Soft Computing*, vol. 70, pp. 604–621, 2018, ISSN: 1568-4946. DOI: https://doi.org/10.1016/j.asoc.2018.06.009. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1568494618303326.

[275] M. Zhang and Y. Chen, "Link prediction based on graph neural networks," *Advances in neural information processing systems*, vol. 31, 2018.

[276] S. Zhang, H. Tong, J. Xu, and R. Maciejewski, "Graph convolutional networks: A comprehensive review," *Computational Social Networks*, vol. 6, no. 1, pp. 1–23, 2019.

[277] X. Zhang, Y. Sun, K. Jiang, C. Li, L. Jiao, and H. Zhou, "Spatial sequential recurrent neural network for hyperspectral image classification," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 11, no. 11, pp. 4141–4155, 2018.

[278] X. Zhang, Y. Sun, J. Zhang, P. Wu, and L. Jiao, "Hyperspectral unmixing via deep convolutional neural networks," *IEEE Geoscience and Remote Sensing Letters*, vol. 15, no. 11, pp. 1755–1759, 2018.

[279] X. Zhang, Y. Sun, J. Zhang, P. Wu, and L. Jiao, "Hyperspectral unmixing via deep convolutional neural networks," *IEEE Geoscience and Remote Sensing Letters*, vol. 15, no. 11, pp. 1755–1759, 2018.

[280] X. Zhang, Y. Sun, J. Zhang, P. Wu, and L. Jiao, "Hyperspectral unmixing via deep convolutional neural networks," *IEEE GRSL*, vol. 15, no. 11, pp. 1755–1759, 2018.

[281] X. Zhang, Y. Sun, J. Zhang, P. Wu, and L. Jiao, "Hyperspectral unmixing via deep convolutional neural networks," *IEEE GRSL*, vol. 15, no. 11, pp. 1755–1759, 2018. DOI: 10.1109/LGRS.2018.2857804.

[282] X.-M. Zhang, L. Liang, L. Liu, and M.-J. Tang, "Graph neural networks and their current applications in bioinformatics," *Frontiers in genetics*, vol. 12, p. 690 049, 2021.

[283] Z. Zhang, "Introduction to machine learning: K-nearest neighbors," *Annals of translational medicine*, vol. 4, no. 11, 2016.

[284] R. Zhao, Y. Hu, J. Dotzel, C. De Sa, and Z. Zhang, "Improving neural network quantization without retraining using outlier channel splitting," in *International conference on machine learning*, PMLR, 2019, pp. 7543–7552.

[285] S. Zhao, Z. Chen, Z. Xiong, Y. Shi, S. Saha, and X. X. Zhu, "Beyond grid data: Exploring graph neural networks for earth observation," *IEEE Geoscience and Remote Sensing Magazine*, 2024.

[286] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun, "Graph neural networks: A review of methods and applications," *AI open*, vol. 1, pp. 57–81, 2020.

[287] F. Zhu, Y. Wang, *et al.*, "Spectral unmixing via data-guided sparsity," *IEEE TIP*, vol. 23, no. 12, pp. 5412–5427, 2014.