Predicting Model Performance for Remote Sensing Hyperspectral Image Classification

Haukur Ísfeld Ragnarsson

PREDICTING MODEL PERFORMANCE FOR REMOTE SENSING HYPERSPECTRAL IMAGE CLASSIFICATION

Haukur Ísfeld Ragnarsson

60 ECTS thesis submitted in partial fulfillment of a
Magister Scientiarum degree in Electrical and Computer Engineering

Advisors
Magnús Örn Úlfarsson
Jón Atli Benediktsson

Examiner
Gabriele Cavallaro

Faculty of Electrical and Computer Engineering
School of Engineering and Natural Sciences
University of Iceland
Reykjavik, May 2021

Predicting Model Performance for Remote Sensing Hyperspectral Image Classification
Predicting Remote Sensing Classification Performance
60 ECTS thesis submitted in partial fulfillment of a M.Sc. degree in Electrical and Computer Engineering

Faculty of Electrical and Computer Engineering
School of Engineering and Natural Sciences
University of Iceland
Dunhagi 5
107, Reykjavik, Reykjavik
Iceland

Telephone: 525 4000

Bibliographic information:
Haukur Ísfeld Ragnarsson, 2021, Predicting Model Performance for Remote Sensing Hyperspectral Image Classification, M.Sc. thesis, Faculty of Electrical and Computer Engineering, University of Iceland.

Printing: Háskólaprent, Fálkagata 2, 107 Reykjavík
Reykjavik, Iceland, May 2021

For Hrönn

Abstract

Hyperspectral image classification has been an active research topic in remote sensing for decades. Every year hundreds of research papers are published that present new classification algorithms and compare them with known algorithms. It is of interest to know to what degree one can predict a classification algorithm's performance using only metadata such as the size of an image being classified, the number of classes and the type of classification method used. In this study, a prediction model was trained by using data on classification accuracy and metadata from 100 research papers on hyperspectral image classification. Linear and nonlinear regression methods were trained and optimized using 5-fold cross-validation and various transformations of the output variable. In addition, an ensemble regression model is trained and optimized using Bayesian optimization. Results show that nonlinearity of the model is important to improve predictions and that an ensemble regression model with a Logit transformation yields prediction with a 3.82% mean absolute error. This research may provide valuable information for remote sensing researchers and can help verify results and develop algorithms.

Útdráttur

Flokkun fjarkönnunarmynda hefur verið stöðugt rannsökuð undanfarna áratugi. Á hverju ári eru gefnir út hundruðir vísindagreina sem fjalla um nýjar aðferðir til flokkunar fjarkönnunarmynda og bera þau saman við þekktar aðferðir. Það er áhugavert að vita hvort hægt sé að spá fyrir um frammistöðu flokkunaraðferða með því að nota eingöngu meta-gögn s.s. stærð myndarinnar sem verið er að flokka, fjölda flokka eða gerð af flokkunaraðferð. Í þessari rannsókn er spálíkan þjálfað með meta-gögnum og flokkunarnákvæmni aðferða úr yfir hundrað birtum vísindagreinum. Línuleg og ólínuleg aðhvarfsgreiningarlíkön eru þjálfuð og bestuð með 5-faldri krossgildingu og með beitingu varpana á spágildi. Niðurstöður sýna að ólínuleika aðhvarfsgreiningarlíkana skiptir höfðumáli í að auka nákvæmni spálíkansins. Að lokum er safnlíkan þjálfað og bestað með notkun Bayes bestunar. Niðurstöður sýna að safnlíkan með Logit vörpun á spágildi nær 3.8% meðaltalsskekkju. Þessi rannsókn getur gefið vísindamönnum í fjarkönnun gagnlega innsýn inn í stöðu rannsókna á flokkun fjarkönnunarmynda og hjálpað til við að meta niðurstöður og þróa nýjar aðferðir.

Contents

List of Figures

List of Tables

Acknowledgements

I would like to thank my thesis advisor, Magnús Örn Úlfarsson, for very enjoyable cooperation and for his support and guidance during my studies. I want to thank Jón Atli Benediktsson for serving on my thesis advisory committee. I also want to thank the faculty at the department of Electrical- and Computer Engineering, particularly Jakob Sigurðsson and Jóhannes R. Sveinsson for their friendliness and advice. I would like to thank my fellow students, members of the signal processing lab, Sveinn, Magnús, Hans, Burkni, Han and Bin for their good company and support. Last but not least, I would like to thank my family and especially my wife Hrönn and my daughter Harpa. Without them none of this would have been possible.

1. Introduction

Remote sensing refers to the practice of capturing images of the Earth from a high altitude such as from a satellite or a drone. The images are taken with specialized cameras, called hyperspectral cameras, that can capture light over a continuous range of wavelengths from shortwave infrared to visible range [1]. These images are referred to as hyperspectral remote sensing images (HSIs). Hyperspectral imaging has applications in various fields such as agriculture where HSIs are used to study crop canopy [2], in surface mineralogy to study and identify minerals on the surface of the Earth [3], and target recognition [4], a key component of surveillance technology when spatial-based image processing techniques are insufficient.

Hyperspectral image classification, which is the process of transforming hyperspectral images into land-cover maps, is fundamental to hyperspectral imaging. Figure 2.2 (a) shows the false-color representation, where three bands are randomly sampled from the spectra to create an RGB image, of the Indian Pines test site image. It shows farmland with different crop fields. Figure 2.2 (b) shows the ground truth labels for the same image, where a person has gone and mapped where the crop fields are and what type of crop and manually labeled them on the image. With this ground truth it's possible to construct an algorithm that can automatically detect which spectrum belongs to which labeled class. In the simplest version, we could use a lookup table and compare each pixel to known spectra. This is difficult for many reasons as discussed in Section 2.1. For the past decades, researchers have developed increasingly more sophisticated methods, ranging from spectral methods [5], through spectral-spatial methods [6], to deep learning [7].

Hyperspectral image classification is a complicated process that depends on the input data, the classification algorithm, and other metadata containing features that describe the classification experiment. In the last decades, many good classification algorithms have been proposed. Considerable effort has gone into comparing such algorithms. The most common approach is to test the algorithm for some benchmark images, while keeping other experimental features, such as how the training and test set are sampled, fixed between experiments [8, 9]. [10] employs statistical meta-analysis to compare pairs of classification algorithms; [11] used scatterplots to investigate the relationship between classification accuracy and various metadata features.

In hyperspectral image classification experiments, metadata is collected that contains features that describe the experiment. This begs the question to what degree this metadata can predict the classification performance. In this study, we ask to what degree metadata can predict the accuracy of a classification algorithm. To help answer this question, we develop a prediction model to predict the classification accuracy based on features typically reported in classification experiments. We have gathered data on performance metrics by sampling published research papers on hyperspectral image classification and logging reported performance on both proposed methods and comparison methods. We believe such a model could help the research community, both for researchers to guide their research and reviewers to evaluate new ideas. We also think the model can be advantageous in applied classification problems where the prediction model can help choose the optimal classification model, given some initial information about the hyperspectral image, by grid searching across the other feature variables.

This thesis is structured as follows: Section 2 describes the data used for this study, the papers sampled, the remote sensing images used, and the metrics used to quantify model performance. Section 3 describes the features we have come up with to label each experiment. These features include preprocessing methods, image metadata, method categorization, and anything possibly relevant to find a good prediction model. It also covers the prediction methods used for regression in the experiments and the metrics used to evaluate the results. Section 4 covers regression analysis on the dataset and an interpretation of results as well as issues that must be addressed. It also covers the results of linear vs nonlinear regression and the final prediction model used and the results from experiments. Finally, Section 5 offers discussion and conclusions.

2. Data

2.1. Hyperspectral Images

A regular color image contains three spectral bands, red, green, and blue, that are captured from the visible wavelength range through lens exposure. Hyperspectral cameras are specialized cameras that can collect hundreds of spectral bands for a single image. These are known as the spectra of an image. HSIs contain the spectra at each pixel at the imaging scene. Figure 2.1 shows how this spectra looks for a hyperspectral image of the Indian Pines test site in northern Indiana, USA. It shows the mean reflectance value of each pixel for every spectral band. We can see the different reflectance values at each wavelength. It is commonplace to make corrections for the water absorption regions. Water absorption is particularly strong around the 1400 nm and 1900 nm region. For Indian Pines, these spectral bands are removed from any processing as can be seen in 2.1. Different materials on the ground have different reflectance at different wavelengths. Therefore, each material has a unique spectral signature that can be detected from the HSIs. In a perfect scenario we could look at the spectrum of each pixel and compare with spectrum collected in a controlled setting. However, remote sensing images are subject to a lot of disturbance. When light enters Earth's atmosphere it scatters off different particles in the atmosphere when the particles are smaller than the wavelength of the light. This phenomenon is known as Rayleigh scattering and is what makes the sky blue at daytime and orange-reddish in the evening. There are also other phenomena at work that cause more scattering and polarization effects to the light. There's also absorption effects where different particles, such as water or carbon dioxide, partially absorp light at different wavelengths. When light finally hits the Earth it is reflected back up through the atmosphere, exposed to the same effects, before finally hitting the sensors of a hyperspectral camera. This, along with measurement errors, causes a great deal of noise and uncertainty in the spectra of each pixel. Additionally, depending on the resolution of the image, there may be several materials present in a single pixel, each one reflecting differently at different wavelengths. There is a whole field of research on hyperspectral unmixing [12] dedicated to techniques to separate these spectral signatures into their constituent parts. All of these factors make it difficult to accurately classify the contents of a hyperspectral image.

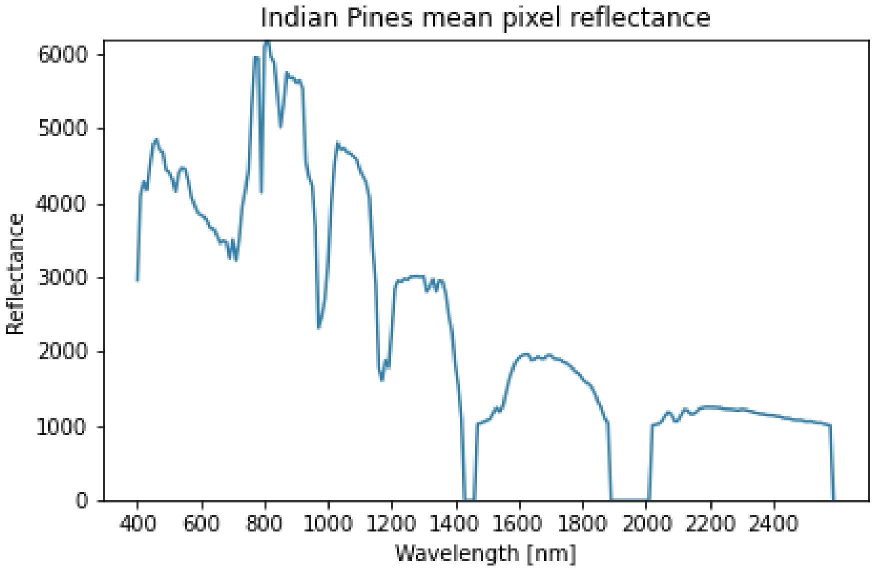


Figure 2.1: Indian Pines mean pixel reflectance.

2.1.1. Indian Pines

The Indian Pines dataset was captured by the AVIRIS sensor [13] over the Indian Pines test site in Northwest Indiana. It is a 145×145 pixel image with 224 spectral bands in the $400 - 2400$ nm wavelength range. 24 bands that cover the water absorption region are removed, leaving 200 spectral bands. The data has also been corrected for atmospheric distortion. The ground truth has 16 labeled classes. Figure 2.2 shows the false color representation and ground truth labels for Indian Pines. Table 2.1 shows the class labels and number of samples.

2.1.2. Pavia University

The Pavia University data was captured by the ROSIS sensor over the University of Pavia in northern Italy. It is a 610×340 pixel image with 103 spectral bands. The ground truth has 9 labeled classes. Figure 2.3 shows the false color representation and ground truth labels for Pavia University. Table 2.2 shows the class labels and number of samples.

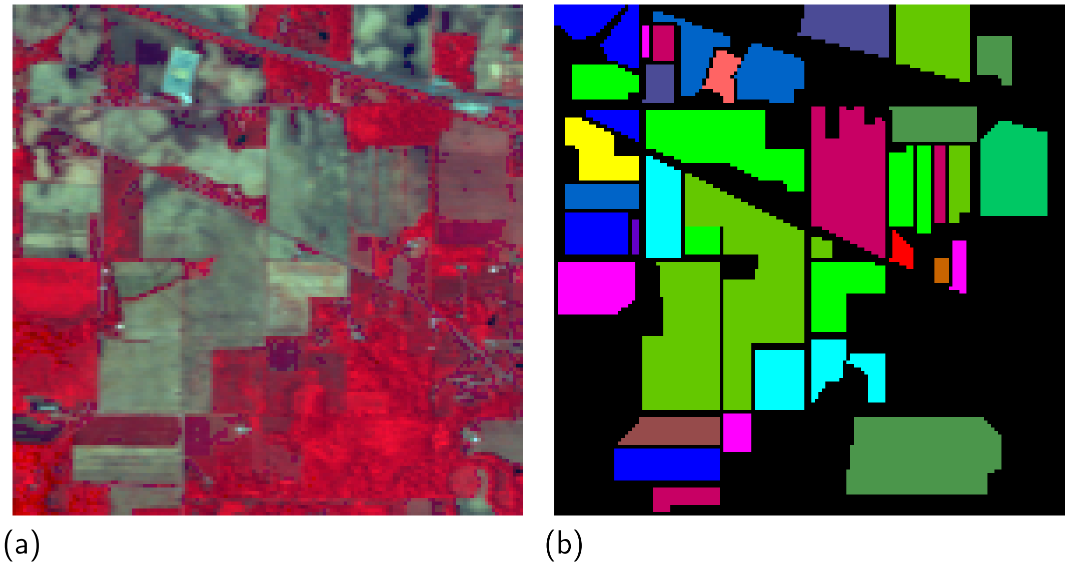


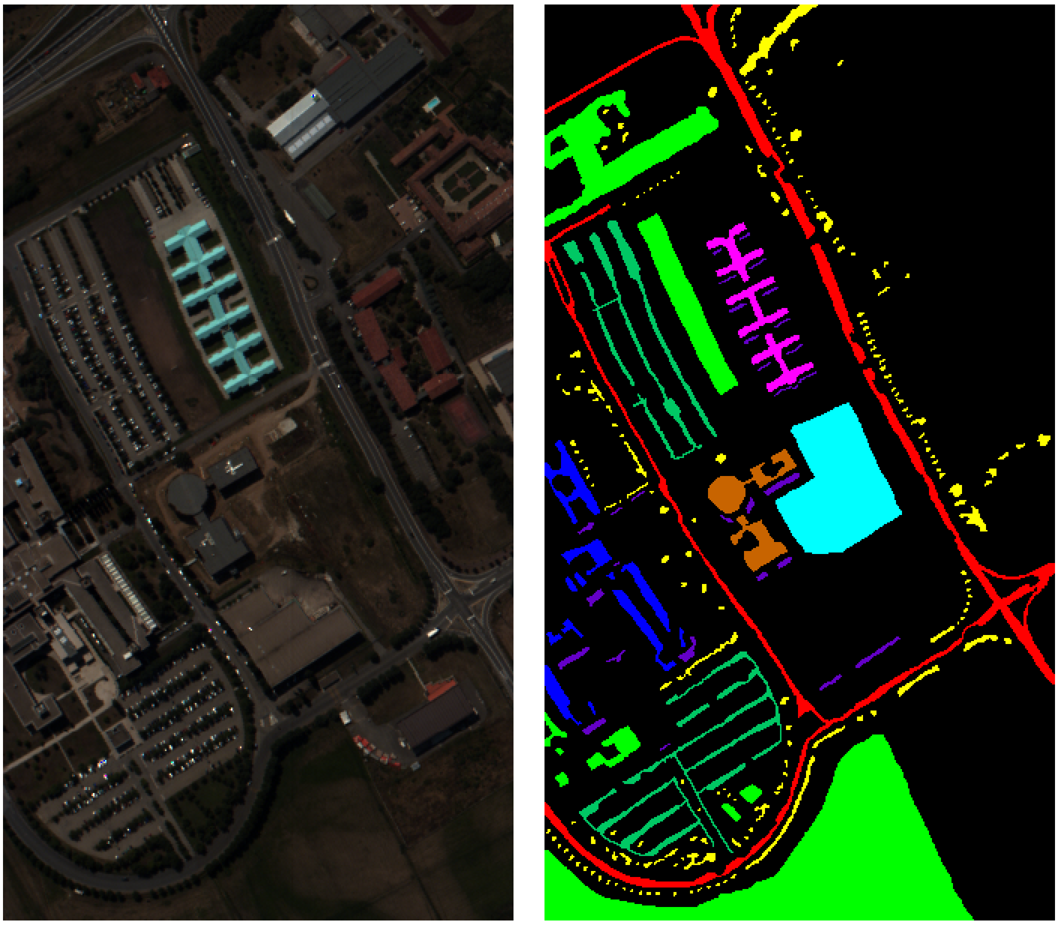
Figure 2.2: Indian Pines. (a) False color representation. (b) Ground truth labels

#	Class	Samples
1	Alfalfa	46
2	Corn-notill	1428
3	Corn-mintill	830
4	Corn	237
5	Grass-pasture	483
6	Grass-trees	730
7	Grass-pasture-mowed	28
8	Hay-windrowed	478
9	Oats	20
10	Soybean-notill	972
11	Soybean-mintill	2455
12	Soybean-clean	593
13	Wheat	205
14	Woods	1265
15	Buildings-Grass-Trees-Drives	386
16	Stone-Steel-Towers	93
	Total	**10249**

Table 2.1: Class labels for Indian Pines.

2.1.3. Pavia Centre

The Pavia Centre data was captured by the ROSIS sensor over the center of Pavia in northern Italy. It is a 1096 $\times$ 1096 pixel image with 103 spectral bands. The



(a) (b)

Figure 2.3: Pavia University. (a) False color representation. (b) Ground truth labels

#	**Class**	**Samples**
1	Asphalt	6631
2	Meadows	18649
3	Gravel	2099
4	Trees	3064
5	Painted metal sheets	1345
6	Bare Soil	5029
7	Bitumen	1330
8	Self-Blocking Bricks	3682
9	Shadows	947
	Total	**42776**

Table 2.2: Class labels for Pavia University.

ground truth has 9 labeled classes. Figure 2.4 shows the false color representation and ground truth labels for Pavia Centre. Table 2.3 shows the class labels and number of samples.

#	**Class**	**Samples**
1	Water	824
2	Trees	820
3	Asphalt	816
4	Self-Blocking Bricks	808
5	Bitumen	808
6	Tiles	1260
7	Shadows	476
8	Meadows	824
9	Bare Soil	820
	Total	**7456**

Table 2.3: Class labels for Pavia Centre.

2.1.4. Salinas Valley

The Salinas Valley dataset was captured over Salinas Valley, California, by the AVIRIS sensor. It is a 512×217 image with 224 spectral bands. 20 spectral bands belonging to the water absorption regions are discarded, leaving 204 spectral bands. The ground truth has 16 labeled classes. Figure 2.5 shows the false color representation and ground truth labels for Salinas Valley. Table 2.4 shows the class labels and number of samples.

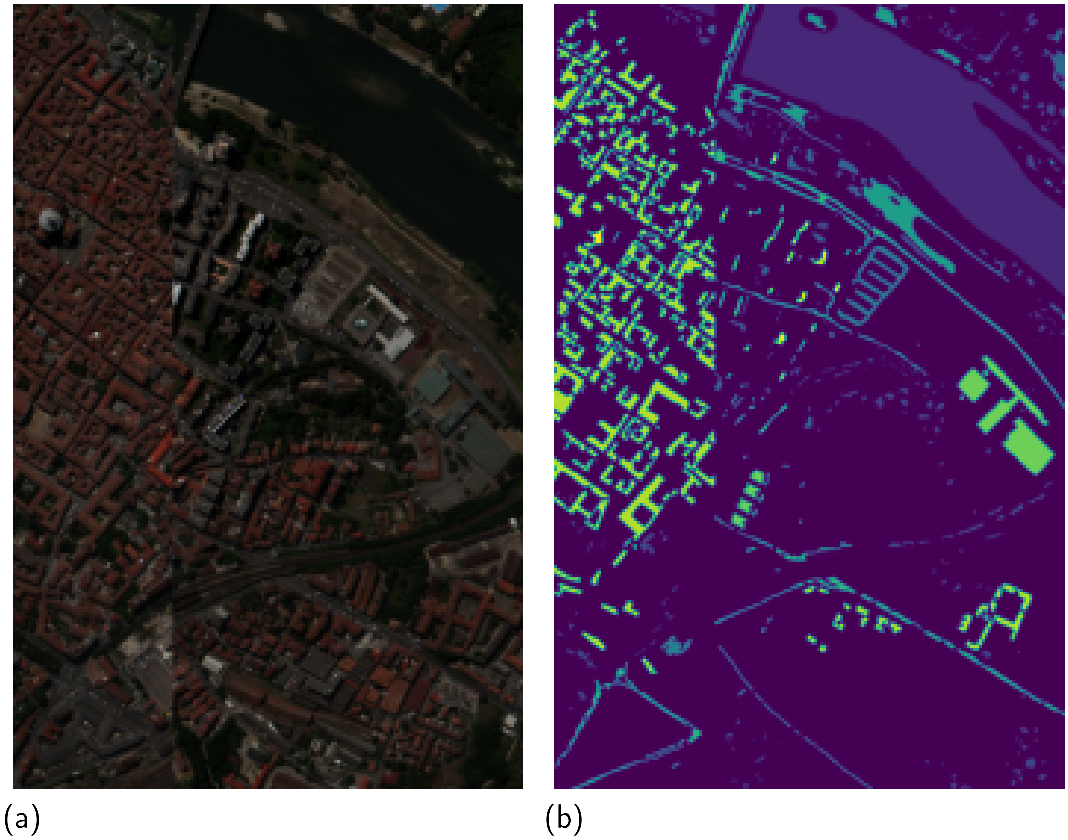


Figure 2.4: Pavia Centre. (a) False color representation. (b) Ground truth labels

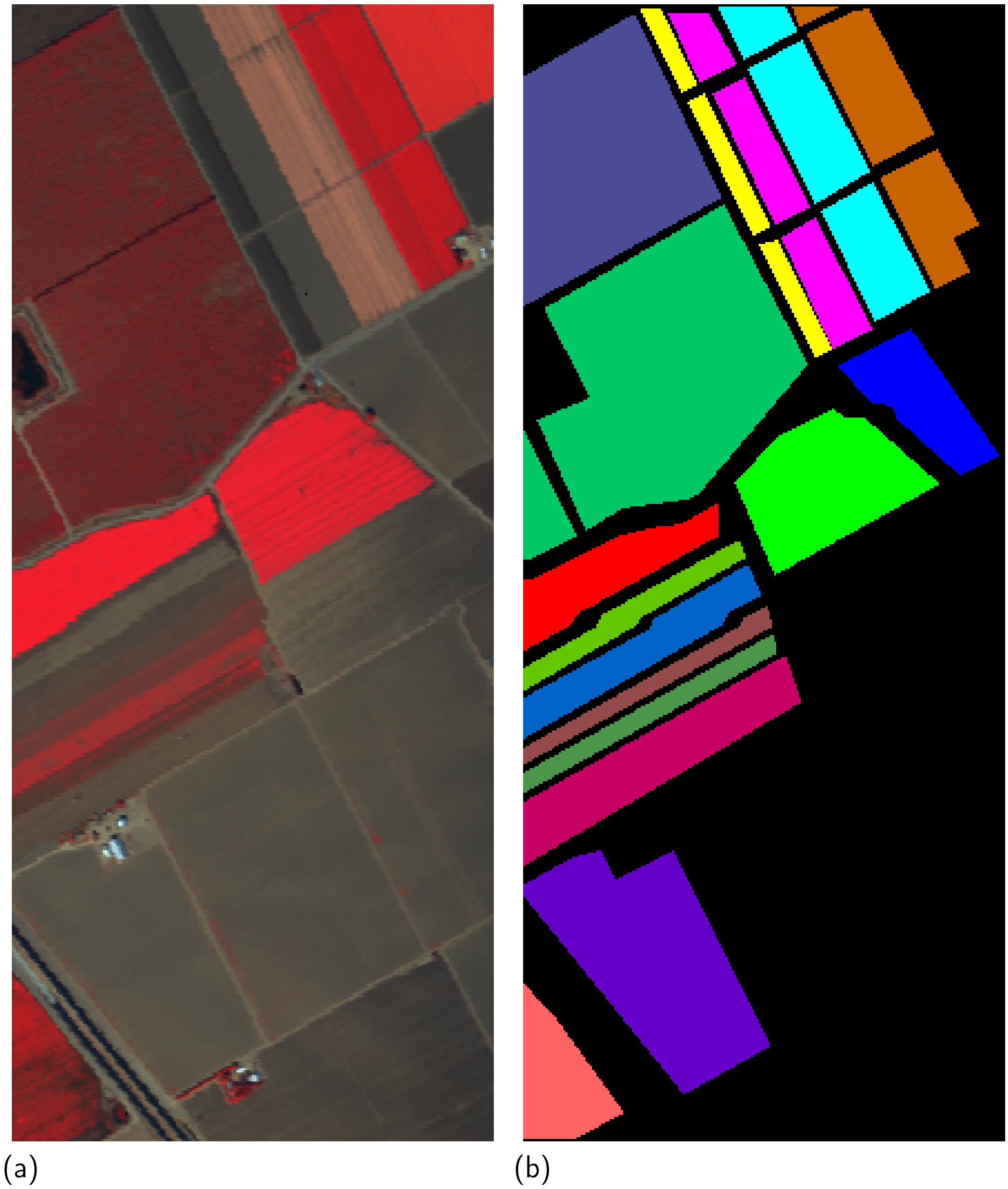


Figure 2.5: (a) Salinas Valley false color image. (b) Salinas Valley ground truth.

#	Class	Samples
1	Brocoli green weeds 1	2009
2	Brocoli green weeds 2	3726
3	Fallow	1976
4	Fallow rough plow	1394
5	Fallow smooth	2678
6	Stubble	3959
7	Celery	3579
8	Grapes untrained	11271
9	Soil vinyard develop	6203
10	Corn senesced green weeds	3278
11	Lettuce romaine 4wk	1068
12	Lettuce romaine 5wk	1927
13	Lettuce romaine 6wk	916
14	Lettuce romaine 7wk	1070
15	Vinyard untrained	7268
16	Vinyard vertical trellis	1807
	Total	**54129**

Table 2.4: Class labels for Salinas Valley.

2.1.5. Salinas-A Scene

Salinas-A Scene is a small sub-scene of the Salinas Valley image. It is 86×83 pixels and is located within the same scene. The ground truth has 6 labeled classes. Figure 2.6 shows the false color representation and ground truth labels for Salinas-A Scene. Table 2.5 shows the class labels and number of samples.

#	Class	Samples
1	Brocoli green weeds 1	391
2	Corn senesced green weeds	1343
3	Lettuce romaine 4wk	616
4	Lettuce romaine 5wk	1525
5	Lettuce romaine 6wk	674
6	Lettuce romaine 7wk	799
	Total	**5348**

Table 2.5: Class labels for Salinas-A scene.

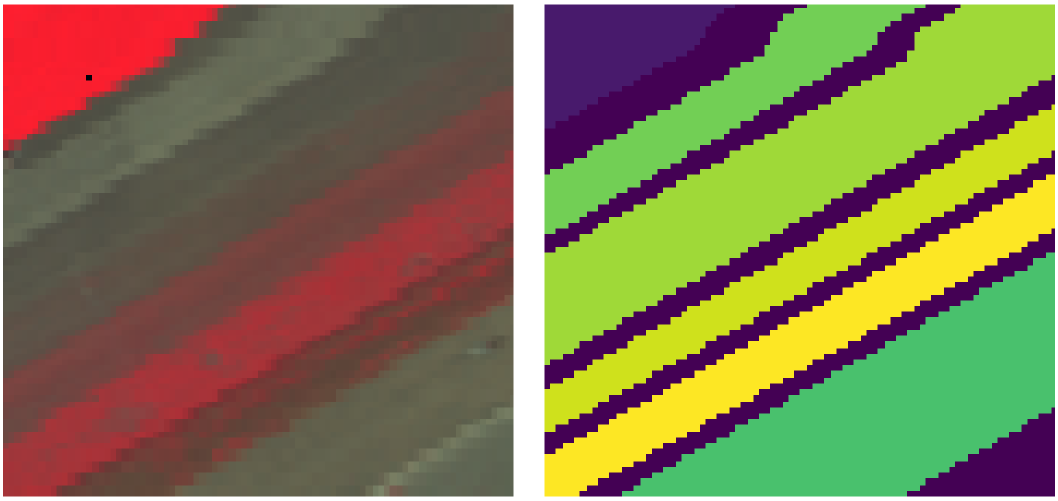


Figure 2.6: (a) Salinas Valley false color image. (b) Salinas Valley ground truth.

2.1.6. Botswana

The Botswana dataset was captured by the NASA EO-1 satellite [14] over the Okavango Delta in Botswana. It is a 1476×256 pixel image with 242 spectral bands covering the $400 - 2500$ nm wavelength region. Spectral bands that cover the water absorption region were removed, leaving 145 bands. The ground truth has 14 labeled classes. Figure 2.7 shows the false color representation and ground truth labels for Botswana. Table 2.6 shows the class labels and number of samples.

2.1.7. Kennedy Space Center

The Kennedy Space Center data was captured by the AVIRIS sensor [13] over the Kennedy Space Center, Florida. It is a 512×614 pixel image with 224 spectral bands in the $400 - 2500$ nm wavelength region. After removing bands covering the water absorption region, 176 bands are left. The ground truth has 13 labeled classes. Figure 2.8 shows the false color representation and ground truth labels for Kennedy Space Center. Table 2.7 shows the class labels and number of samples.

2.1.8. Houston

The Houston dataset was acquired as part of the 2013 IEEE GRSS Data Fusion Contest. The scene is over the University of Houston and the neighboring area. It

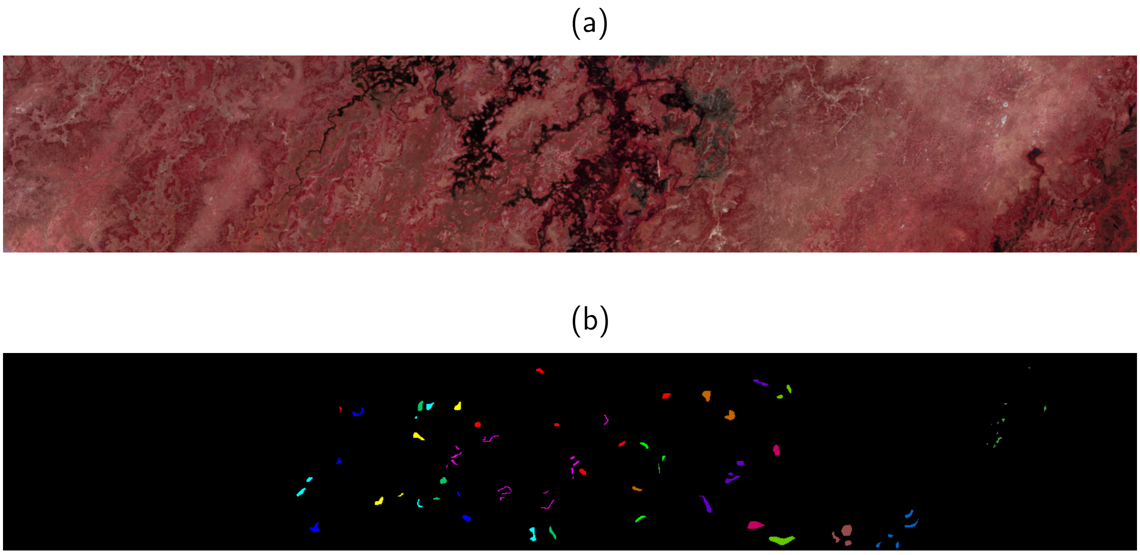


Figure 2.7: (a) Botswana false color image. (b) Botswana ground truth.

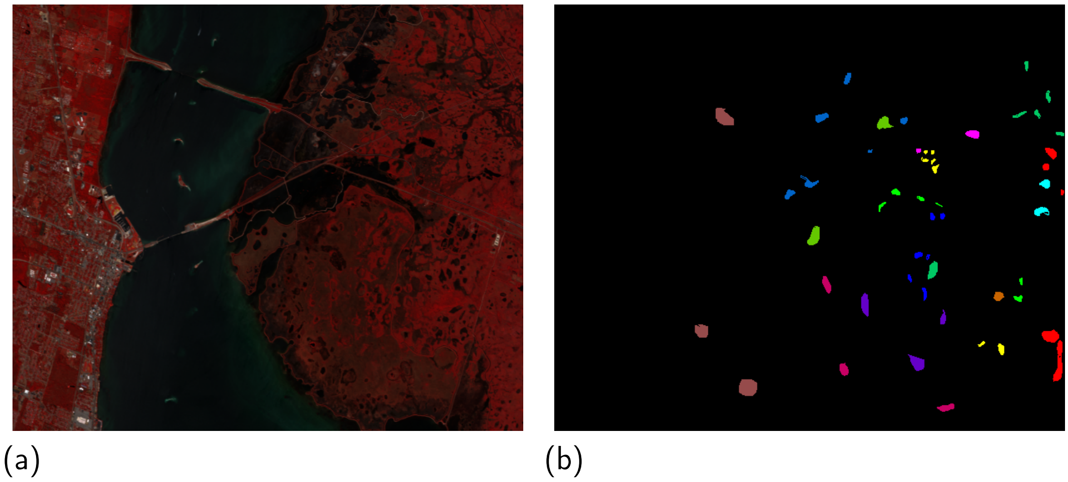


Figure 2.8: (a) Kennedy Space Center false color image. (b) Kennedy Space Center ground truth.

#	Class	Samples
1	Water	270
2	Hippo grass	101
3	Floodplain Grasses 1	251
4	Floodplain Grasses 2	215
5	Reeds 1	269
6	Riparian	269
7	Firescar 2	259
8	Island interior	203
9	Acacia woodlands	314
10	Acacia shrublands	248
11	Acacia grasslands	305
12	Short mopane	181
13	Mixed mopane	268
14	Exposed soils	95
	Total	**3248**

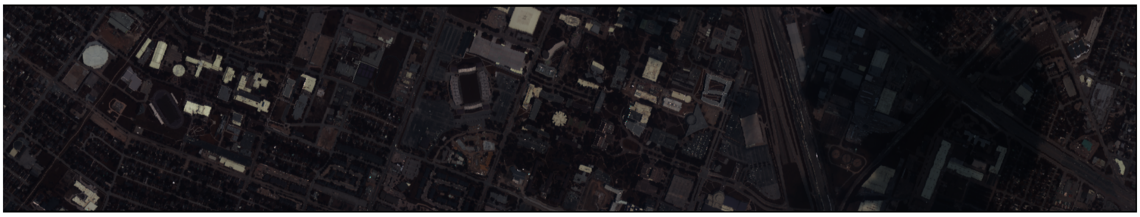
Table 2.6: Class labels for Botswana.

#	Class	Samples
1	Scrub	761
2	Willow swamp	243
3	Cabbage palm hammock	256
4	Cabbage palm/oak hammock	252
5	Slash pine	161
6	Oak/broadleaf hammock	229
7	Hardwood swamp	105
8	Graminoid marsh	431
9	Spartina marsh	520
10	Cattail marsh	404
11	Salt marsh	419
12	Mud flats	503
13	Water	927
	Total	**5211**

Table 2.7: Class labels for Kennedy Space Center.

is a 349×1905 pixel image of 144 spectral bands. The ground truth has 15 labeled classes. Figure 2.9 shows the false color representation and ground truth labels for Houston. Table 2.8 shows the class labels and number of samples.

(a)



(b)

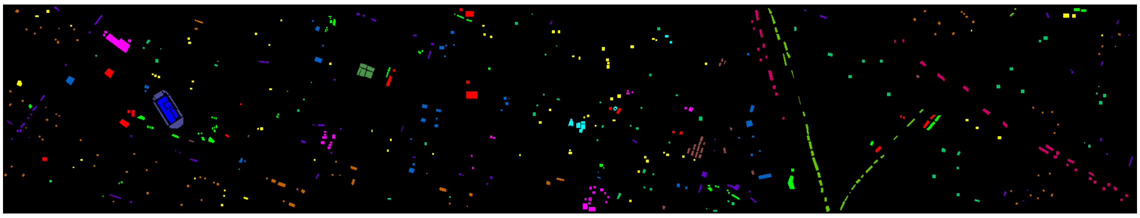


Figure 2.9: (a) Houston false color image. (b) Houston ground truth.

#	Class	Samples
1	Healthy grass	1251
2	Stressed grass	1254
3	Synthetic grass	697
4	Trees	1244
5	Soil	1242
6	Water	325
7	Residential	1268
8	Commercial	1244
9	Road	1252
10	Highway	1227
11	Railway	1235
12	Parking lot 1	1233
13	Parking lot 2	469
14	Tennis court	428
15	Running track	660
	Total	**15029**

Table 2.8: Class labels for Houston.

2.1.9. Washington DC Mall

The Washington DC Mall dataset was captured by the HYDICE sensor [15] over Washington DC. It is a 1208×307 pixel image with 210 spectral bands in the $400-$

2400 nm wavelength region. After removing bands covering the water absorption region, 191 bands are left. The ground truth has 6 labeled classes.

2.2. Research Papers

One hundred papers regarding the classification of hyperspectral remote sensing images using a combination of processing techniques and classification algorithms were analyzed in this study. The papers are published between 2002 and 2021 and cover six main categories of classification algorithms as discussed in Section 3.1. The sampling of papers by year and sampling split by method category is summarized in Figure 2.2 and Figure 2.3, respectively. Each article presents a new classification method and reports performance metrics for the proposed method and comparison methods. We have also run some of the techniques ourselves on different hardware and for different sizes of training sets when the code is available. The total number of reported performance metrics is 2964 from 686 unique methods. Some papers are overview papers that cover a wide range of known methods, e.g., [16–19] while others present a novel approach to the problem, e.g., [20, 21]. Techniques and algorithms have a significant impact on the outcome. They can be broken down into categories, as presented in [16]. These categories were used to sample data from ieeexplore.org. To restrict the search for papers to remote sensing, we searched for papers on hyperspectral classification and used keywords for each category to find relevant papers. For example, papers on mathematical morphology-based classification were found with the search query ""remote sensing" AND hyperspectral classification AND morphology". For each set of results, a few of the most cited papers and some additional papers that are more recent or more relevant for the topic were sampled. Additionally, papers that have open source code available were found by searching for code repositories on github.com tagged with "hyperspectral-image-classification". This search yields several repositories and relevant papers.

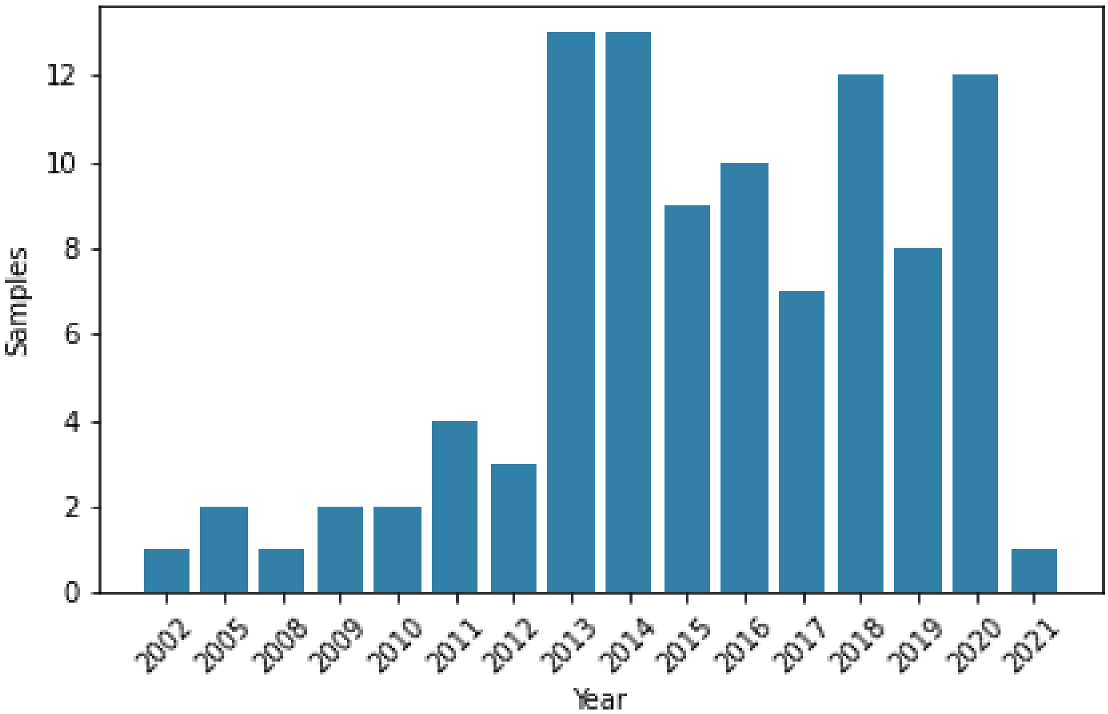


Figure 2.10: Overall sampling of papers by year.

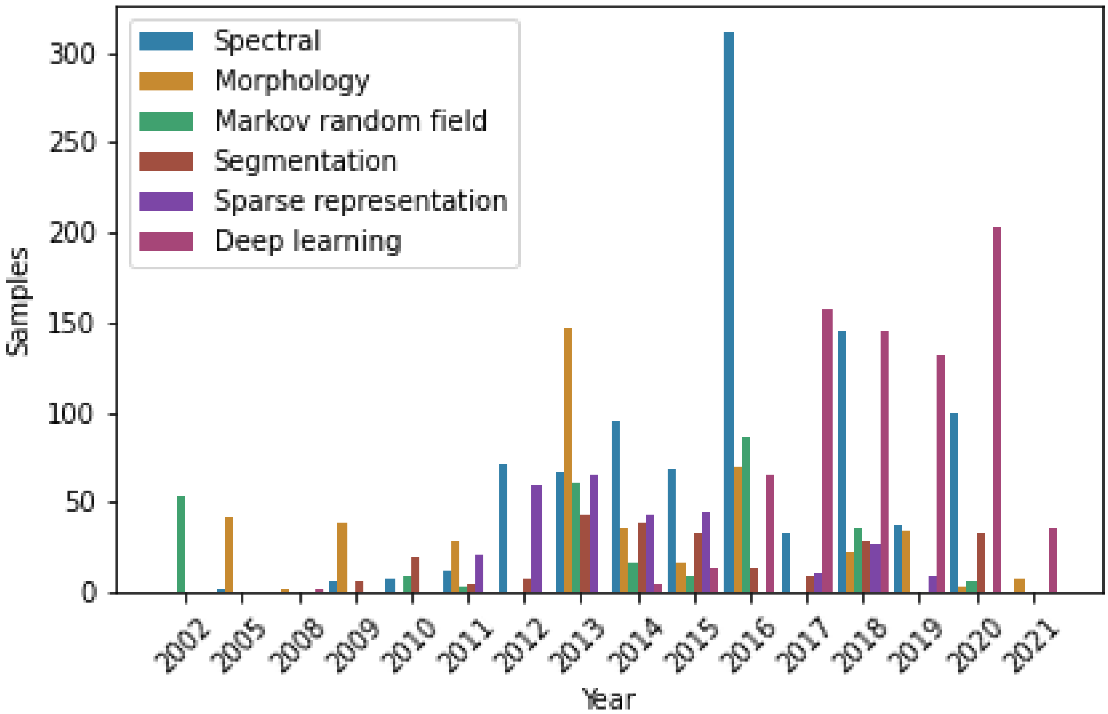


Figure 2.11: Overall sampling of methods by year, split by method category.

3. Methods

3.1. Method Categories

It can be helpful to sort methods into different categories. The categories presented in [16] are used, namely: mathematical morphology-based, Markov random fields, segmentation, sparse representation, and deep learning-based as well as spectral classifiers. These categories do an excellent job of differentiating the underlying mechanism of each method. Each method is labeled with one of these categories (as MC) and treatment encoding, sometimes called dummy encoding, is used on the categories when regression is done.

3.1.1. Spectral Classifiers

As discussed in section 2.1, hyperspectral images are 3D data cubes where every pixel in an image is a vector of spectral bands. Spectral classifiers only take the spectral vector as input and ignore any information about the surrounding pixels. Support Vector Machines (SVM) [22] and Random Forest (RF) classifiers [23] are well-known algorithms that are commonly used as spectral classifiers. Most of the research papers sampled for this thesis present metrics for SVM or RF classifiers as a baseline with which to compare their proposed methods. Other methods presented in our collection of papers describe not so much a novel classifier algorithm but rather a novel preprocessing technique before classifying the data using an SVM. For example, [24] presents an SVM classifier with a clever sampling technique designed to boost the classifier performance. To overcome the limitations of imbalanced sampling between classes when training an SVM classifier, the support vectors themselves are sampled rather than the training data. This is a different approach to solving a recurrent problem in hyperspectral image classification using what is otherwise a very well-known algorithm. Other papers present methods based on a minimum spanning forest [25] or random walkers [26] to do spectral classification.

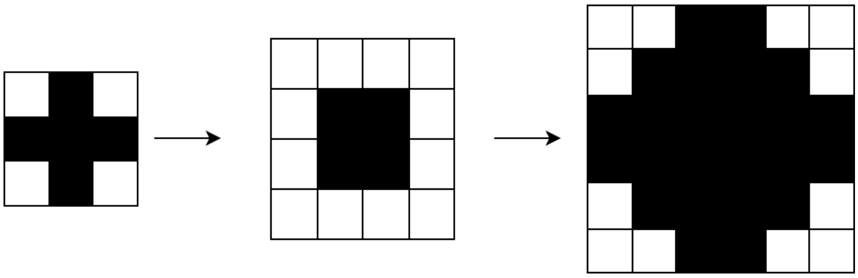


Figure 3.1: Mathematical morphology dilation operation.

3.1.2. Mathematical Morphology

Mathematical morphology-based classifier methods work on pre-processing the data before applying classification algorithms. By using morphological operators, a morphological profile is constructed of each sample and these profiles are then classified. Common morphological operators are erosion and dilation operators. A structural element B is defined, maybe a 3x3 square or a 3x3 cross as an example. This structural element is then applied to a set of pixels A such that the erosion $\ominus$ of A by B is the set of all pixels C such that B, translated by C, is fully contained in A or

$$A \ominus B = \{C|(B)_C \subseteq A\}\,. \tag{3.1}$$

By contrast, dilation $\oplus$ of A by B is the set of all pixels C such that B, translated by C, overlaps with at least one element of A or

$$A \oplus B = \{C|(B)_C \cap A \neq \varnothing\}\,. \tag{3.2}$$

Figure 3.1 shows how this operation works. These operations can be performed on a hyperspectral image in either 2D or 3D and can serve to highlight certain structures in the image while eliminating others. There are many possible structural elements and other morphological operations that can be used.

Extended Multi-Attribute Profiles

Attribute Profiles are created by applying different morphological filters in sequence to an image. The Extended Attribute Profile is created from the first few principal components of the spectral vector instead of the entire spectrum. Finally, Multi-Attribute Profiles are a combination of different Attribute Profiles. Extended Multi-Attribute Profiles (EMAPs) are heavily used in hyperspectral classification applications as reported in [27–32] together with sparsity, decision rules, Bilateral Filters and Canonical Polyadic decomposition to improve the classification. [33–35] use Extended Attribute Profiles along with neural networks while [36–41] use many different approaches such as using Adaptive Manifold Filters fused with spatial correlation features, Local Binary Patterns, second-order statistics for feature extraction, Independent Component Analysis instead of Principal Component Analysis (PCA) for the extended profiles, the standard deviation to build the profiles, and wavelets to extract features. Finally [42], studies the influence different morphological operators have on classification accuracy.

3.1.3. Markov Random Fields

Markov random fields (MRF) are joint probability distributions over a set of variables defined as an undirected graph. In the case of hyperspectral images, for every pixel vector $\mathbf{x}_i$ the probability of a label y_i is defined as

$$p(y_i|\mathbf{x}_i) = \prod_{c \in C} \phi_c(\mathbf{x}_{ic}) \tag{3.3}$$

where C is the neighbourhood of pixel i and $\phi_c(\mathbf{x}_{ic})$ is a pairwise potential between pixel i and neighbourhood pixel c, sometimes called a clique. This pairwise potential can be defined in a number of ways to create different MRF models. These MRF models are then combined with classifiers such as SVM and Gaussian Mixture Models as shown in [43], belief propagations or tree reweighted message passing as shown in [44] or convolutional neural networks as shown in [45] for a conditional random field. It has been shown that MRF modeling is especially helpful for the classification of high-resolution hyperspectral images.

MRF are routinely used as postprocessing of classification maps as a kind of segmentation method, so the final classification label is a function of the label of the adjacent pixels, defined by the clique used with the MRF model. [43, 46–51] use this method with Gaussian Mixture Models, SVMs and Logistic Regression as classifiers. [44] uses a Conditional Random Field instead of an MRF. [52–59] combine MRFs with many different methods such as unary energy functions, homogenous areas priors, Expectation-Maximization, Object-oriented segmentation, Bayesian contextual classification, Loopy Belief Propagation, and sparsity.

3.1.4. Segmentation

Segmentation-based classifiers use segmentation algorithms to partition an image into large homogenous regions. A classifier such as SVM is used to classify the regions, usually pixelwise. The final result is obtained with majority voting within the region. This approach is excellent for eliminating outlier errors, where a classifier will wrongly classify a pixel that belongs to a larger topological structure.

Segmentation Techniques

There are several techniques to segment an image. One is based on edge detection where discontinuities or edges are found in the images and these edges are treated as the separation between regions. [60] uses an Edge-Preserving Filter to find the edges while [61] uses the Watershed algorithm. Another method is to use graph-based segmentation. In a graph, each node in the graph constitutes a pixel and the connections between nodes are weights given by some defined similarity function or another metric. The graph is then cut based on some rule or threshold for the connection. Nodes that remain connected after the graph cut constitute a region. [61, 62] use a Minimum Spanning Forest to create the final segmentation map while [63] uses graphs to construct a superpixel map. Regions can also be iteratively optimized by applying initial markers and iteratively grow regions from the markers as is done in [61, 64] with Expectation-Maximization to cluster regions together. [65, 66] use Particle Swarm Optimization and Hierarchical Optimization, respectively, to cluster regions.

3.1.5. Sparse Representation

Sparse representation is the method of representing a signal where only a few parameters are nonzero. As an example, a Fourier transform of a sine wave is represented as two delta impulses at the sine wave frequency. One in the positive frequency spectrum and one in the negative. All other frequencies are zero. Therefore the Fourier transform is a form of sparse representation of the sine wave. A sparse representation is a coding scheme where a signal is represented as a dictionary made up of basic elements, which may or may not be orthogonal. To find a sparse representation one must minimize the Frobenius norm of the residual between the signal and the dictionary multiplied by the representation, under some sparsity constraint. In terms of a hyperspectral image we have a dictionary $\mathbf{D} = [\mathbf{D}_1, \cdot, \mathbf{D}_C] \in \mathbb{R}^{d \times N}$, where d is the number of spectral bands, C is the number of classes and N is the total number of elements in the dictionary. The dictionary is drawn from the training pixels, each column for each class. The sparse representation can then be achieved by optimizing for

$$\underset{\mathbf{r}\in\mathbb{R}^N}{\operatorname{argmin}}\|\mathbf{x}-\mathbf{Dr}\|_2 \tag{3.4}$$

where $\mathbf{x}$ is the signal and $\mathbf{r}$ is subject to $\|\mathbf{r}\|_0 \leq S_0$ where S_0 is some sparsity constraint. This model is then expanded to include spatial context and can be solved using orthogonal matching pursuit algorithms. Classification models in our dataset, using sparse representation algorithms, are described in [67–76].

3.1.6. Deep Learning

The last category, which has seen the most research activity in recent years, is deep learning-based methods. Deep learning has seen an explosion in the last ten years, in research and applications, across a wide range of industries. The application of deep learning algorithms in hyperspectral classification uses artificial neural networks to learn a representation of the data through many layers of neurons and returns a probability map of the classification possibilities. By training the network through backpropagation of the error, the network adapts itself to learn the most optimal representation and can correctly classify new data.

Artifical Neural Network

An artificial neural network was first introduced in 1943 by McCulloch et al. [77] and has been an active research topic ever since. For a long time, they were deemed too computationally expensive to be useful at anything until the late 1990's and early 2000's when advances in hardware made them feasible for solving commercial applications, resulting in a burst of research interest. A neural network consists of layers of neurons. Each neuron connects to every neuron in the next layer. The input to a neuron is a weighted sum of the output of the previous layer. This weighted sum is then passed through an activation function before being output to the next layer. Figure 3.2 illustrates how this works. The output of the last layer is the predicted variable of the neural network, a probability map in the case of classification and in the case of regression, a single neuron with a linear activation function. The error between the predicted value and the training sample is then backpropagated through the network by calculating the gradient of the cost function with respect to the weights of each layer. The weights can then be optimized through an optimization function such as stochastic gradient descent. All the parameters of the network, the number of layers, the number of neurons in each layer, the choice of the cost

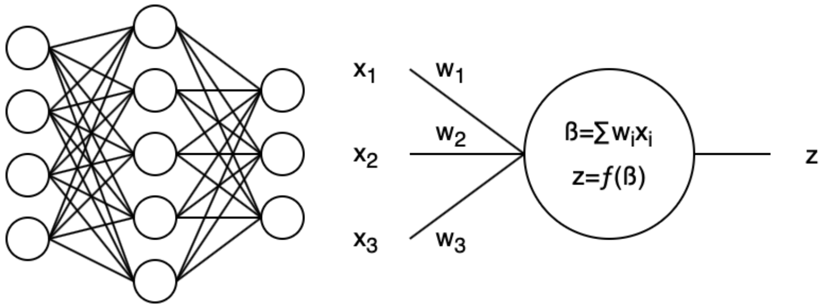


Figure 3.2: Artificial Neural Network

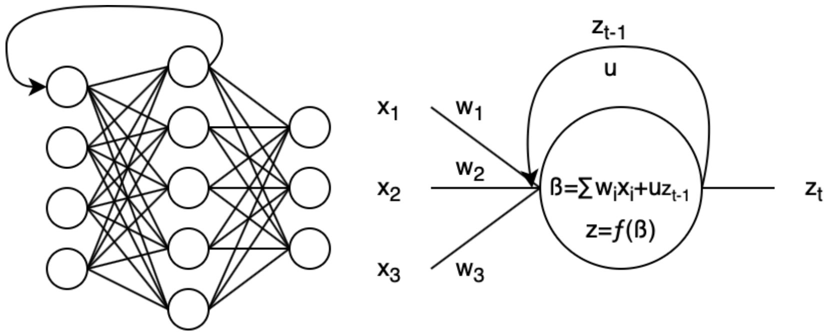


Figure 3.3: Recurrent Neural Network

function, and the optimization function can be selected and tuned. This makes neural networks very flexible models, and although they can be computationally expensive, they can be very powerful in a multitude of applications. However, because they are so flexible, they are difficult to tune.

Recurrent Neural Network

Recurrent neural networks (RNN) work in exactly the same way as artificial neural networks with one addition. In each hidden layer of an RNN, the output may be fed back to the input of a previous layer, as shown in Figure 3.3. This means that while training, when processing one sample through the network, the output of a hidden layer from a previous sample can be added to the input of the current layer. This means that an RNN has a memory component which makes them suitable for processing time-series data. In [78], an RNN is used for the classification of a hyperspectral image by feeding each spectral band into the network. The hidden state of the network becomes the calculation of the previous spectral band. The classification label is finally found by looping through all the spectral bands in a pixel. [56] combines segmentation with RNNs for classification.

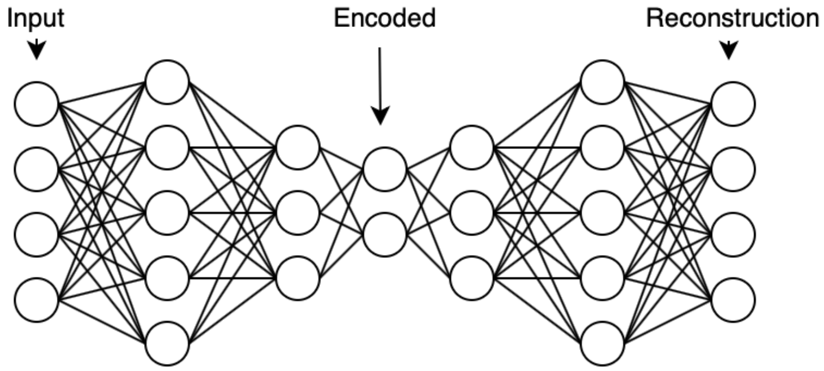


Figure 3.4: Autoencoder

Autoencoder

An Autoencoder is a type of unsupervised neural network. Consisting of an encoder, a decoder and a hidden representation, an autoencoder trains a neural network to predict its own input. The encoder of the network can be neural network architecture. It encodes the input to a latent subspace of reduced dimension. The encoded output is then fed into the decoder. An example of how an autoencoder works is shown in Figure 3.4. The decoder can also be any neural network architecture. Often it is a mirror of the decoder. The output of the decoder is a reconstruction of the input. The error function for the autoencoder then incorporates the error between the input and the reconstruction for backpropagation. An autoencoder can be used for many things such as feature reduction, sparsity or denoising. [79] uses an autoencoder based on a deep convolutional neural network for hyperspectral image classification.

Convolutional Neural Networks

The most commonly used neural network for image processing is a convolutional neural network (CNN), which learns a series of convolutional filters to perform the classification. The problem with using a regular artificial neural network for image processing is the high dimensionality of an image. In a regular image classification problem, such as the classic 'cat' or 'not cat' classification, one training sample consists of a whole image. A 100×100 pixel image unrolled in a single vector means the input vector is 10000 features. Even with an Artificial Neural Network with only a single hidden layer of 100 neurons, that equals a million weights. That's just for a single spectral band. Most images are a lot larger than 100×100 pixels and many spectral bands, 3 for an RGB image, hundreds for a hyperspectral image. This many weights makes for a prohibitive computational complexity. To solve this problem, the CNN was developed. Instead of a hidden layer with millions of

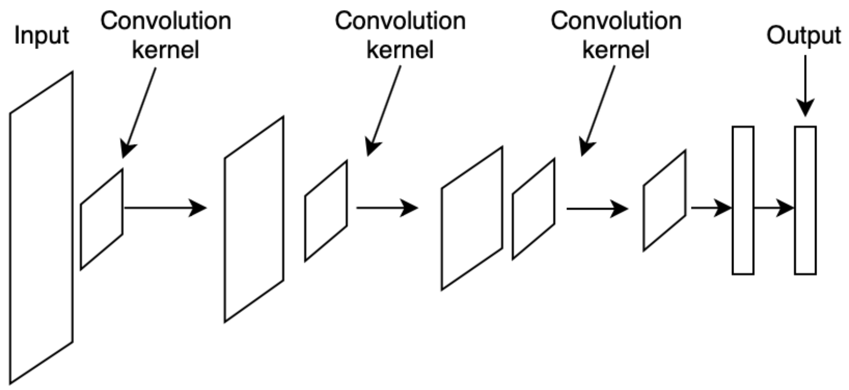


Figure 3.5: Convolutional Neural Network

weights between layers, you can have a convolutional kernel with only a few dozen or hundred of weights. This kernel is then convolved with the image and adjusted through backpropagation. Once the image has been reduced sufficiently in dimension from convolution, it can be unrolled and connected to a normal neural network layer to produce a classification map. This type of architecture is shown in Figure 3.5. Although the dimensionality problem is not as severe in hyperspectral image classification since each training sample is only a single pixel, not an entire image, it is still an issue nonetheless. CNNs are routinely used in hyperspectral image classification to incorporate spatial information as well as spectral information.

[20, 45, 80–109] all use CNNs for hyperspectral image classification. The architecture of the networks vary tremendously with 2D and 3D convolution kernels [83], residual architectures [86, 91, 92, 94, 103], General Adversarial Networks [107] and combining them with MRFs [45] and Attribute Profiles[100] to name a few.

Deep Belief Network

A Deep Belief Network (DBF) is a little different from a regular feedforward neural network. Each layer of a DBF is composed of a Restricted Boltzmann Machine (RBM). Instead of feedforward prediction and error backpropagation an RBM is an undirected graph. An RBM is trained to the log probability of a sample x with respect to the weights W between nodes or

$$\underset{W}{\operatorname{argmax}}\{\mathbb{E}[log(P(x))]\} \tag{3.5}$$

IN a DBF, each layer of RBM is chained together, so the hidden layer of one RBM is the input to the next. When training a DBF, the RBMs can be pretrained with unsupervised training and then the whole network can be finetuned using error backpropagation between them. [110] showed that DBFs that are trained in this way can achieve better results than deep neural networks of comparable size. [111, 112] use DBFs for hyperspectral image classification.

3.2. Features

3.2.1. Image

As described in Section 2.2 we have performance metrics for several different hyperspectral images. These images have different characteristics depending on the sensor used to capture them and the scene that is captured. These include the number of classes to classify (Ncl), the spatial resolution of the image (Res), the number of spectral bands (Nsb) and the type of scene depicted in the image (Scn), whether it is rural or urban.

3.2.2. Sampling

When performing an experiment, data is sampled into training and test sets. This is usually done in one of two ways. The first is to randomly sample the training set from the entire pool of labeled samples. The second is to sample, either randomly or by hand, a certain percentage, or a fixed number, of samples from each labeled class. The latter method ensures that the training set includes samples from all classes. We, therefore, label each method by their sampling method (Split), either uniformly sampled or sampled per class, as well as the overall number of training and testing samples (Ntr and Nts, respectively).

3.2.3. Preprocessing

Often there is some form of preprocessing performed on the data before classification. It is common practice to scale the features of the data either by normalization or standard scaling. Normalization is done by scaling the pixel values to the interval between zero and one as described in [85]. Standard scaling is done by scaling the pixel values to have zero mean and unit variance as described in [91]. This practice is, however, usually not reported in the papers as is the case in [88] where examination of the source code shows that standard scaling is used.[1] We therefore

[1] https://github.com/KonstantinosF/Classification-of-Hyperspectral-Image

consider it to be standard. The most common method of preprocessing is dimensionality reduction. PCA is often used to reduce the size of the spectral vector from hundreds of values to 10 or 15 pixels as done in[88, 90]. We, therefore, label each method by whether dimensionality reduction (DR) has been done. Sometimes the dimensionality reduction is embedded in the proposed method and it is difficult to determine what number of spectral bands are used for actual classification. In these cases, the number of spectral bands is left unchanged from the number given for the dataset.

3.2.4. Spectral vs Spectral-Spatial

Most hyperspectral image classification methods fall into either the spectral or spectral-spatial method category. Spectral methods utilize only the spectral information of the pixel being classified, while spectral-spatial methods also incorporate the spatial information of the surrounding pixels. We label the methods with this feature (MT).

3.2.5. Other

We have included other possibly significant features that can tell us something about the methods or the experiments that gave the results. The number of experiments performed for averaging (Avg) of the result, whether by cross-validation or not, is one and whether the code behind the experiment is open-sourced or available somehow for evaluation (OSCode). Both of these features indicate whether the result is validated properly and how reliable it is.

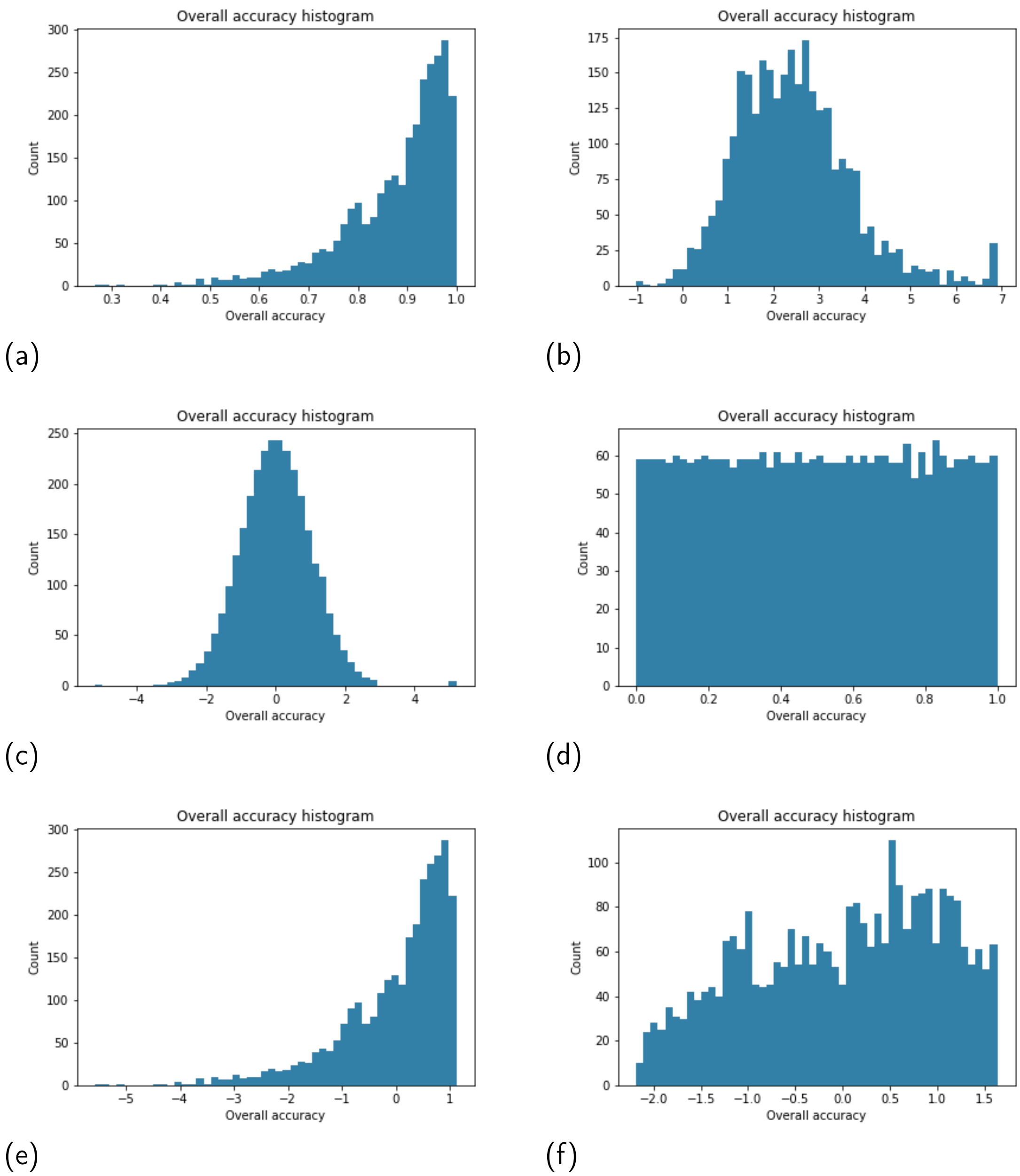


Figure 3.6: Histogram of overall accuracy. (a): No transformation. (b): Logit transformation. (c): Gaussian transformation. (d): Uniform transformation. (e): Standard transformation. (f): Power transformation.

Table 3.1: Summary of Features.

Feature	Label	Encoding
Number of training samples	Ntr	Integer > 0
Number of testing samples	Nts	Integer > 0
Number of classes	Ncl	Integer > 0
Number of spectral bands	Nsb	Integer > 0
Resolution	Res	Float > 0.0
Scene	Scn	Binary
Sampling method	Split	Binary
Dimensionality Reduction	DR	Binary
Method type	MT	Binary
Method category	MC	Treatment coding
Averaging	Avg	Integer > 0
Open source code	OSCode	Binary

3.3. Prediction Methods

3.3.1. Linear Models

Baseline Model

To properly evaluate the performance of prediction models, it is useful to define a baseline model to use for comparison. This model is usually defined in very simple terms and any other model that is put up for consideration should at least perform equally well and, ideally, better than this baseline model. In this project, a baseline model is defined as the mean of the predicted variables in the training set.

Ordinary Least Squares Regression

The model we use for regression analysis is an ordinary least squares regression. The equation for the regression is given as

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \epsilon \tag{3.6}$$

where ϵ is white zero-mean Gaussian noise with variance σ^2,

$$\begin{aligned}
\mathbf{y} =& [y_1, \ldots, y_n]^T \\
\mathbf{X} =& \begin{bmatrix} \mathbf{x}_1^T \\ \vdots \\ \mathbf{x}_n^T \end{bmatrix} \\
\mathbf{x} =& [1, \mathrm{Ntr}, \mathrm{Nts}, \mathrm{Ncl}, \mathrm{Nsb}, \mathrm{Res}, \mathrm{Scn}, \mathrm{Split}, \mathrm{DR}, \mathrm{MT}, \\
& \mathrm{MC2}, \mathrm{MC3}, \mathrm{MC4}, \mathrm{MC5}, \mathrm{MC6}, \mathrm{Avg}, \mathrm{OSCode}]^T \\
\boldsymbol{\beta} =& [\beta_0, \ldots, \beta_{17}]^T
\end{aligned}$$

where $\mathbf{y}$ is the output, $\mathbf{x}$ contains the input variables along with an intercept as the first element, n is the number of training samples and β are the regression parameters. For ordinary least squares regression, equation (3.6) is optimized according to $\hat{\boldsymbol{\beta}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$. The model is fitted to the entire dataset with overall accuracy, average accuracy, and Cohen's Kappa as the output.

Ridge Regression

Ridge regression is the same as ordinary least squares regression but with an added l_2 regularization. Whereas the objective function to be minimized is $\min_{\boldsymbol{\beta}} \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2$ for ordinary least squares, for ridge regression the objective function is

$$\min_{\boldsymbol{\beta}} \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 + \alpha \|\boldsymbol{\beta}\|_2^2 \tag{3.7}$$

Thus the α term puts a penalty on the coefficient, larger coefficients mean a larger error. This helps to reduce the coefficients to the most important independent variables and reduce multicollinearity problems. Ridge regression is often applied when the matrix $\mathbf{X}$ is ill-defined, resulting in multicollinearity when using ordinary least squares regression [113].

Lasso Regression

Lasso regression is the same as ordinary least squares regression but with an added l_1 regularization where the objective function to be minimized is

$$\frac{1}{2N} \min_{\boldsymbol{\beta}} \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 + \alpha \|\boldsymbol{\beta}\|_1 \tag{3.8}$$

where N is the number of training samples. Lasso regression, like Ridge regression, is also used to combat high collinearity by promoting sparsity in the independent variables [114]. This time the α imposes a penalty on the coefficients through the l_1 norm.

Elastic Net

Elastic net is ordinary least squares regression but with a combined l_1 and l_2 regularization where the objective function to be minimized is

$$\frac{1}{2N}\min_{\boldsymbol{\beta}}\|\mathbf{y}-\mathbf{X}\boldsymbol{\beta}\|_2^2+\alpha L\|w\|_1+\frac{1}{2}\alpha(1-L)\|\boldsymbol{\beta}\|_2^2 \tag{3.9}$$

where N is the number of samples and L is a hyperparameter to determine the ratio between l_1 and l_2 regularization. Elastic Net is used for the same purpose as Lasso and Ridge, but is considered to have better performance in certain situations, particularly when the number of independent variables is much greater than the number of observations [115].

Orthogonal Matching Pursuit

Sometimes sparse linear regression is performed such that the objective function to be minimized is $\min_{\boldsymbol{\beta}}\|\mathbf{y}-\mathbf{X}\boldsymbol{\beta}|_2^2$ subject to $\|\boldsymbol{\beta}\|_0 < k$. Such a sparsity constraint is NP-hard to solve. Orthogonal Matching Pursuit[116] is an iterative algorithm to find a subset of β that fullfills the sparsity constraint. Let $\mathbf{X}_{t_i}$ be independent variable of column t of iteration i and $\mathbf{r}_i$ be the residual of the linear regression for iteration i. Let c be a set, containing the variable columns t from each iteration. The algorithm works iteratively in three steps with the initial condition $\mathbf{r}_0 = \mathbf{y}$ and $c = \emptyset$,

1. Solve the maximization problem $\max_t\|\mathbf{X}_t^T\mathbf{r}_i\|$ and add t to a set c.

2. Project the variables $\mathbf{X}$ onto a linear space spanned by the subspace $\mathbf{X}(c_i)$ with $\mathbf{P}_i = \mathbf{X}(c_i)(\mathbf{X}(c_i)^T\mathbf{X}(c_i))^{-1}\mathbf{X}(c_i)^T$. This is an orthogonal projection, hence the name.

3. Update $\mathbf{r}$ according to $\mathbf{r}_i = (\mathbf{I}-\mathbf{P}_i)\mathbf{y}$. Stop if a stopping criterion is reached, such as if the sparsity constraint is broken or a threshold for the residual r is reached.

When this algorithm converges, a set c has been found that gives us a sparse solution $\boldsymbol{\beta}(c)$ that approximates the sparse linear regression solution. This method is usually employed when the number of variables in a regression model exceeds the number of available samples.

3.3.2. Nonlinear Models

Support Vector Regression

In Support Vector Regression, the objective function changes from the previous linear models. We know want to minimize $\frac{1}{2}\|\boldsymbol{\beta}\|^2$ subject to $\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 < \epsilon$. This also means that the final model ignores a large subset of the dependent data that falls outside the error margin ϵ. A regularizing term is therefore added where a small variable ξ, named slack variable, is added to the error margin such that the objective function becomes $\frac{1}{2}\|\boldsymbol{\beta}\|^2 + C\sum^N \xi_n$ subject to $\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 < \epsilon + \xi$ where C is now a new hyperparameter that can tuned to better fit the model. To further augment this method, the predicted variables can be projected with a kernel function to a higher dimensional space. These projections are often nonlinear projections, but they can simplify the optimization problem [117].

Random Forest Regression

Random Forest Regression is the method of creating an ensemble of decision trees on a random subset of the data. Decision trees for regression, called Regression trees, are constructed iteratively by splitting the values of the independent variables at the point that minimizes the error criterion. The point of the split is called a node, which splits into two branches. This is how a tree is built until the number of observations at each node exceeds a specified limit, at which point the node turns into a leaf and the value of the leaf is the average of the observations belonging to that leaf. For regression with multiple independent variables, this process is performed for each variable and the variable that has the lowest error, according to the error criterion, becomes the root node, and so on. When the tree has exhausted all variables and has completed splitting every node, the tree is built. For each tree in the forest, randomly select samples from the training set and train a decision tree on the subset. After training, prediction can be made by running through all the decision trees in the forest and averaging the prediction results [118]. Additionally a modified version of Random Forest, called Extremely Randomized Trees (Extra-Trees), can be used. This models differs from Random Forest in two distinct ways. First, rather than building each tree by randomly sampling the training set, each tree is built using the entire training set. Second, rather then splitting nodes by searching for the most optimum split, each node is split randomly. This can result in increased bias and lowered variance compared to a Random Forest. Extra-Trees are also computationally more efficient than Random Forests due to not searching for the optimal node split for each node of every tree.

Multilayer Perceptron

A Multilayer Perceptron model is another name for an artificial neural network, as described in section 3.1.6 on deep learning. When used for regression, the activation function for the final layer is a linear activation function. This allows the output to be a continuous variable.

Adaboost

Adaboost is a method that, like Random Forest, combines an ensemble of decision trees. However, unlike Random Forest, Adaboost deliberately creates the ensemble out of shallow decision trees, or stumps. These stumps are an example of a weak learner. Also, unlike Random Forest, the Adaboost algorithm iteratively adjusts the relative importance of each training sample as well as the voting power of each stump. As the training progresses, the ensemble focuses more and more on the more important variables and each training set is likely to be comprised of more important samples. This makes Adaboost more tuned to scenarios that are harder to predict, making it more robust to outliers than Decision Trees or Random Forests [119].

Histogram Gradient Boosting

Gradient boosting is a similar method to Adaboost. The difference is that this time the trees that make up the ensemble can be larger than stumps, as in Adaboost, and they are trained to optimize the residual between the predicted and the mean of the target values. Histogram Gradient Boosting is a modified version of Gradient Boosting that uses histograms of feature values and not the feature values themselves when constructing the decision trees [120].

K Nearest Neighbors

A K Nearest Neighbors regression model is one of the simplest outlined in this project. For every input value, find the k nearest neighbors in the training set, through some distance metric. The predicted value is simply the average of these points. Sometimes the average can be weighted by the distance to the neighboring points. The algorithm can be tuned through hyperparameters such as the number of neighbors k, the distance metric and whether to weigh the averaging of predictions.

Ensemble Model

It is possible to construct an ensemble model from any of the prediction methods discussed previously. In this case, the final prediction is a weighted average of the prediction of all the methods in the ensemble. The same optimization techniques that are used for parameter search for each individual model can be used to decide which models should be in the ensemble and how they should be weighted.

3.4. Metrics

We evaluate model performance based on the mean absolute error (MAE) between predicted and actual outcomes, as shown in (4).

$$\mathrm{MAE} = \frac{\sum_{i=1}^{n} |y_i - \hat{y}_i|}{n}. \tag{3.10}$$

We also evaluate the model based on goodness of fit by measuring the coefficient of determination, also known as R^2 for linear models. For linear regression, the R^2 is a measure of the correlation between the dependent and independent variables as given by (5)

$$R^2 = \frac{\sum(\hat{y}_i - \overline{y})^2}{\sum(y_i - \overline{y})^2}. \tag{3.11}$$

where $\hat{y}$ is the predicted value, y is the observed value, and $\overline{y}$ is the mean of all observations. For perfect predictions, R^2 equals one. For nonlinear models, the R^2 score is insufficient because, as (3.11) suggests, R^2 is a fraction of the variance of the predictions and total variance of the observations. The assumption is that the difference between these variances equals the variance of the residual error, and so the R^2 score is always between 0 and 1. This does not hold for nonlinear models, and therefore, the R^2 score for nonlinear models is not reported.

We also evaluate the t-statistic for each dependent variable which is the ratio between the coefficient and the standard error as given by (3.12) where σ is the standard deviation of the entire population and n is the sample size.

$$t_i = \frac{\hat{\beta}_i \sqrt{n}}{\sigma(\hat{\beta}_i)} \tag{3.12}$$

We also want to evaluate the null hypothesis for each of the dependent variables by examining their p-values. The p-values can be defined as the probability that an observed t-value is more extreme than the t-statistic T, given the null hypothesis H_0. This relationship is shown in (3.13)

$$p_i = Pr(|t| \geq T|H_0). \tag{3.13}$$

If an independent variable's p-value falls below a certain threshold α, it is interpreted as the variable having a statistically significant impact on the output of the prediction. This interpretation must be made with care though, since a low p-value is not automatically a rejection of the null hypothesis but instead serves as an indication as to which independent variables have the most influence in our regression model. We will consider this interpretation in our discussion of results.

4. Results

4.1. Datasets

There are nine different datasets in our collected data. A breakdown of these datasets is shown in Table 4.1. These images differ considerably in both resolution and content with some displaying farmland and others urban landscapes, some large homogenous regions and others scattered morphology.

Table 4.1: Breakdown of hyperspectral images.

Image	Sensor	No. of classes	No. of spectral bands	Resolution (m)	No. of samples	OA	AA	Kappa
Indian Pines	AVIRIS	16	220	20	816	83.54 ± 13.22	83.77 ± 14.18	0.82 ± 0.15
Salinas Valley	AVIRIS	16	220	3.7	317	92.12 ± 6.73	92.12 ± 9.09	0.91 ± 0.08
Salinas A-scene	AVIRS	6	220	3.7	23	94.0 ± 6.2	91.86 ± 7.34	0.89 ± 0.11
Pavia University	ROSIS	9	103	1.3	973	86.46 ± 10.43	88.97 ± 9.62	0.83 ± 0.14
Pavia Centre	ROSIS	9	103	1.3	290	95.14 ± 5.38	93.79 ± 6.48	0.96 ± 0.02
Botswana	NASA EO-1	14	145	30	94	92.53 ± 5.58	95.62 ± 4.78	0.95 ± 0.05
Houston	NCALM	15	144	2.5	81	79.36 ± 10.28	80.69 ± 10.4	0.77 ± 0.11
Kennedy Space Center	AVIRIS	13	176	18	209	89.41 ± 9.69	8514 ± 12.31	0.87 ± 0.1
Washington DC Mall	HYDICE	6	191	2.5	161	92.57 ± 6.59	91.44 ± 5.24	0.93 ± 0.04

4.2. Target Variable

There are three target variables that are usually reported for hyperspectral image classification. Overall accuracy, average accuracy and Cohen's Kappa. Most of the results and discussion in this study is focused on overall accuracy. The results for average accuracy and Cohen's Kappa are given in appendix.

4.2.1. Overall Accuracy

Overall accuracy is calculated as the percentage of accurately classified samples over the entire test set. For each sample i in the test set X_i we generate a prediction $\hat{y}_i$.

Overall accuracy is then defined as

$$OA = \frac{1}{N}\sum_{i=1}^{N}[\hat{y}_i = y_i] \tag{4.1}$$

where N is the number of samples in the test set and y_i is the i-th labeled sample in the test set.

4.2.2. Average Accuracy

Average accuracy is the average of each class's percentage of accurately classified samples. For each sample i in the test set X_i we generate a prediction $\hat{y}_i$. Average accuracy is then defined as

$$AA = \frac{1}{C}\sum_{c=1}^{C}\frac{1}{N_c}\sum_{i=1}^{N_c}[\hat{y}_i = y_i] \tag{4.2}$$

where N_c is the number of samples in the test set that belong to class c and y_i is the i-th labeled sample in class c in the test set.

4.2.3. Cohen's Kappa

Cohen's Kappa is a measure of how overall accuracy compares to expected accuracy. To calculate Cohen's Kappa we define two classifiers. We first define p_0 as

$$p_0 = \frac{1}{N}\sum_{c=1}^{C}[n_{c1} = n_{c2}] \tag{4.3}$$

where n_{c1} is the number of times classifier 1 predicted class c and n_{c2} of the number of times classifier 2 predicted class c. p_0 is therefore measuring the agreement between two classifiers. We next define p_e as

$$p_e = \frac{1}{N^2}\sum_{c=1}^{C}n_{c1}n_{c2} \tag{4.4}$$

where, again, n_{c1} is the number of times classifier 1 predicted class c and n_{c2} of the number of times classifier 2 predicted class c. Here p_e is measuring the probability of a chance agreement between classifier 1 and classifier 2. Finally we calculate Cohen's Kappa as

$$\kappa = \frac{p_0 - p_e}{1 - p_e} \tag{4.5}$$

We see now that if the observed agreement between classifiers p_0 is high, κ approaches 1. If p_0 is low we are measuring the impact of observed agreement over change agreement p_e which approaches zero as p_0 approaches p_e. When p_0 is less than p_e, κ is negative and we have agreement between classifiers that is worse than random.

4.3. Linear Regression Results

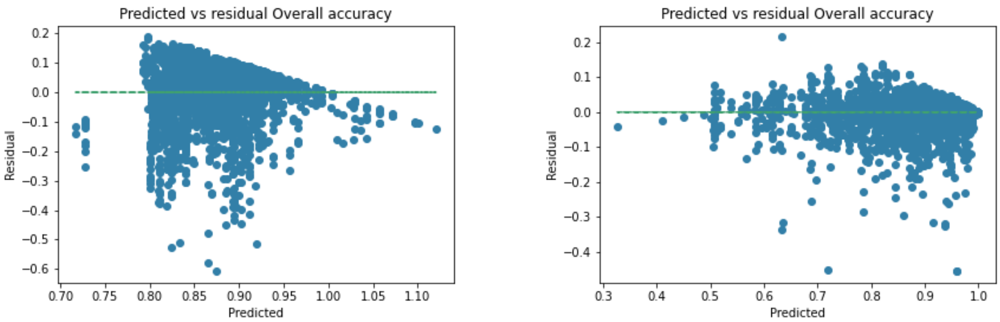
Results from regression analysis are shown in Table 4.2. For overall accuracy the mean absolute error is 0.074 or 7.4% and an R^2 score of 0.183. The low R^2 score along with low p-values for several predictors indicates that the regression model can somewhat capture the overall trend in the data but has high variance in prediction errors. For more precise predictions a higher R^2 score is desired and to achieve that, few techniques are possible, such as finding another predictor, transforming the data or trying a non-linear model.

Figure 4.1 (a) shows the residual between for the linear regression model used in the regression analysis for overall accuracy. There is considerable heteroskedasticity in the residual. A linear least-squares regression model finds a linear combination of features that optimizes the squared distance between the model and all the sample points. One of the assumptions of a good linear model is that the error, or residual, between the model and sample points has near-constant variance across all prediction values. This is known as homoskedasticity. Conversely, heteroskedasticity is changing variance in residuals across prediction values. As seen in Figure 4.1 (a), our regression model has a similar variance in residuals on the lower end of predictions. As we move up in prediction values, residual variance increases on the low side and decreases on the high side. Another assumption of linear regression is that the prediction values are unbounded in each direction. However, because our model is a regression on accuracy values which are bounded at 0 and 100% this also violates this assumption. This is shown in Figure 4.1 (a), where residuals on the high side trend towards zero. We can also see that some prediction values are outside the bounds. What this shows is that a linear regression model is insufficient to deal with the data we have collected and to get a more accurate prediction model we must try other models as will be discussed below.

Table 4.2: Regression analysis results.

	Overall Accuracy		Average Accuracy		Cohen's Kappa	
	MAE	R^2	MAE	R^2	MAE	R^2
	0.074	0.183	0.073	0.227	0.091	0.194
Feature	T	P-value	T	P-value	T	P-value
Intercept	119.670	0.0	99.783	0.0	77.582	0.0
Ntr	10.946	0.0	8.510	0.0	10.989	0.0
Nts	-5.674	0.0	-4.084	0.0	-2.036	0.042
Ncl	-7.114	0.0	-5.786	0.0	-4.299	0.0
Nsb	2.639	0.008	-1.796	0.073	5.28	0.0
Res	-0.718	0.473	-2.258	0.024	0.266	0.791
Scn	-1.027	0.305	-5.259	0.0	0.096	0.923
Split	-0.4	0.689	5.798	0.0	4.875	0.0
DR	1.575	0.115	-0.435	0.663	-2.032	0.042
MT	13.074	0.0	12.004	0.0	9.123	0.0
MC2	3.145	0.002	1.031	0.303	0.555	0.579
MC3	3.365	0.001	-0.021	0.983	1.326	0.185
MC4	1.817	0.069	-0.162	0.871	0.167	0.868
MC5	6.759	0.0	5.244	0.0	5.802	0.0
MC6	5.228	0.0	2.297	0.022	2.418	0.016
Avg	1.019	0.308	2.287	0.022	1.6	0.11
OSCode	2.241	0.025	3.248	0.001	2.763	0.006

It is desired to increase the precision in our prediction and stay within the bounds of the dependent variables. In order to do this, a number of transformations, shown in Figure 3.6, on the dependent variable are performed and run through a number of linear and nonlinear models. The linear models are tuned using a cross-validated grid search and all experiments are performed through a 5-fold cross-validation and report the average results with standard deviation. The nonlinear models are optimized using auto-sklearn, which uses Bayesian optimization to find a good model for the data. The optimization step is run for 20 minutes on a GPU with 5-fold cross-validation. Table 4.3 and Table 4.4 show the results of this experiment and illustrate a few key points. Applying transformations does not seem to significantly improve results and there is no remarkable difference between different models. The only significant factor seems to be the nonlinearity of the models.



(a) (b)

Figure 4.1: (a) Linear Regression residuals for Overall Accuracy. (b) Ensemble Regression residuals for Overall Accuracy.

Table 4.3: Overall accuracy prediction results for linear regression models. Values are given as mean absolute error [%] with standard deviation.

	Transformations					
	No	Logit	Standard	Power	Gaussian	Uniform
Baseline	8.46 ± 0.34	8.06 ± 0.34	8.46 ± 0.34	8.08 ± 0.35	8.03 ± 0.35	8.03 ± 0.35
Linear	7.43 ± 0.33	6.96 ± 0.3	7.43 ± 0.33	7.02 ± 0.34	6.92 ± 0.32	7.01 ± 0.33
Ridge	7.43 ± 0.33	6.97 ± 0.31	7.43 ± 0.33	7.03 ± 0.34	6.93 ± 0.32	7.02 ± 0.34
Lasso	7.74 ± 0.38	7.2 ± 0.36	7.74 ± 0.38	7.39 ± 0.36	7.2 ± 0.35	7.36 ± 0.34
OMP	7.78 ± 0.41	7.23 ± 0.38	7.78 ± 0.41	7.39 ± 0.37	7.22 ± 0.38	7.32 ± 0.36
ElasticNet	7.74 ± 0.38	7.21 ± 0.34	7.75 ± 0.38	7.38 ± 0.35	7.21 ± 0.35	7.36 ± 0.34
Average	$\mathbf{7.62 \pm 0.37}$	$\mathbf{7.11 \pm 0.34}$	$\mathbf{7.63 \pm 0.37}$	$\mathbf{7.24 \pm 0.35}$	$\mathbf{7.1 \pm 0.34}$	$\mathbf{7.21 \pm 0.34}$

Table 4.4: Overall accuracy prediction results for nonlinear regression models. Values are given as mean absolute error [%] with standard deviation.

	Transformations					
	No	Logit	Standard	Power	Gaussian	Uniform
Ensemble	3.93 ± 0.17	3.82 ± 0.18	3.93 ± 0.18	3.84 ± 0.18	3.85 ± 0.19	3.84 ± 0.18
SVR	5.32 ± 0.19	5.17 ± 0.22	5.22 ± 0.2	5.44 ± 0.19	5.14 ± 0.21	5.19 ± 0.25
MLP	4.62 ± 0.16	4.39 ± 0.12	4.38 ± 0.09	4.29 ± 0.12	4.3 ± 0.21	4.67 ± 0.26
Random forest	4.0 ± 0.18	3.88 ± 0.15	4.0 ± 0.18	3.89 ± 0.16	3.91 ± 0.17	3.89 ± 0.16
Ada boost	4.28 ± 0.17	4.17 ± 0.14	4.28 ± 0.18	4.16 ± 0.12	4.23 ± 0.19	4.16 ± 0.13
Histogram gradient boosting	4.21 ± 0.18	4.19 ± 0.15	4.21 ± 0.18	4.25 ± 0.14	4.1 ± 0.17	4.39 ± 0.16
KNN	4.12 ± 0.21	4.02 ± 0.21	4.12 ± 0.21	4.02 ± 0.19	4.04 ± 0.22	4.01 ± 0.19
Average	$\mathbf{4.43 \pm 0.18}$	$\mathbf{4.3 \pm 0.17}$	$\mathbf{4.37 \pm 0.17}$	$\mathbf{4.34 \pm 0.15}$	$\mathbf{4.29 \pm 0.2}$	$\mathbf{4.39 \pm 0.19}$

4.4. Nonlinear Regression Results

An experiment is run using the prediction model described in Section 3.3.2 to predict overall accuracy, using 5-fold cross-validation to evaluate results. The model is optimized using auto-sklearn, running for two hours on Google Colab with a Tesla T4 GPU. The ensemble model from this optimization consists of one Extra Tree

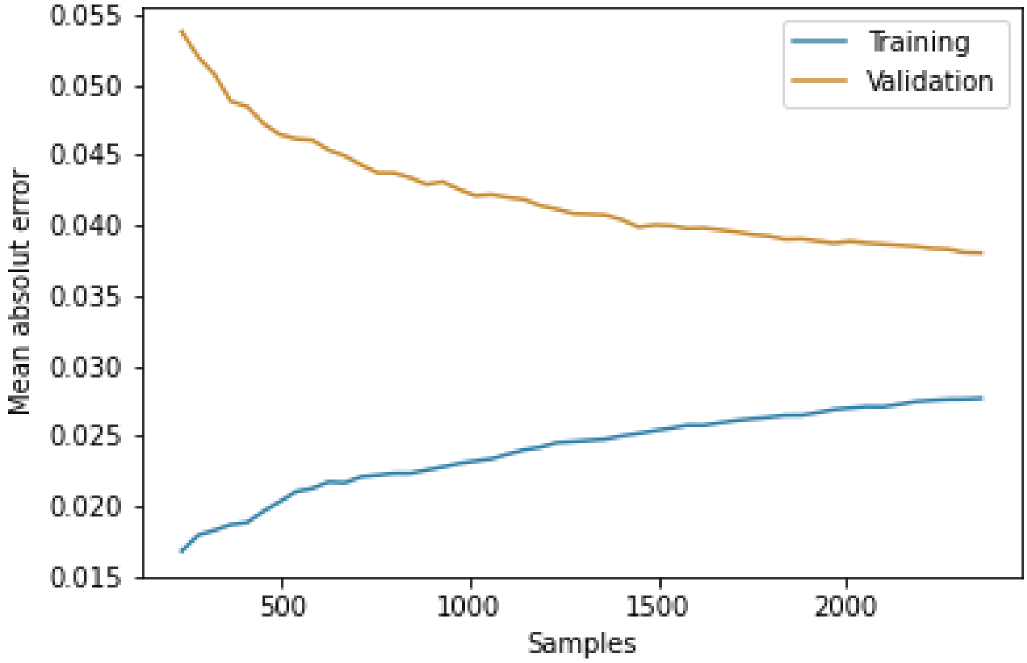
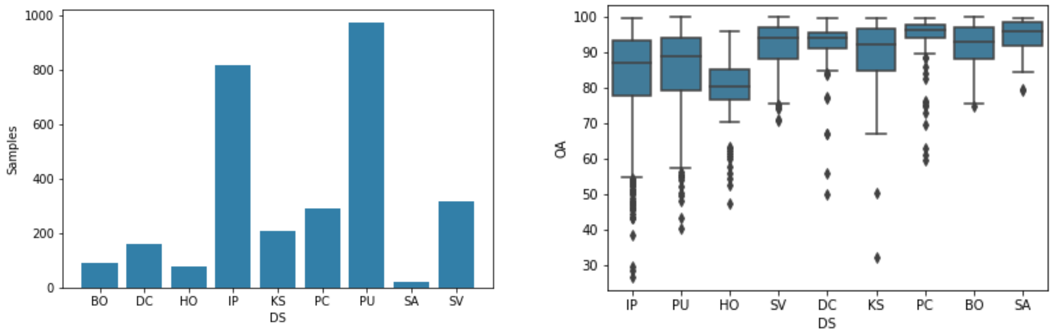


Figure 4.2: Learning Curve for Ensemble Regressor.

Regressor, five Random Forest Regressors and one Support Vector Regressor model where the Extra Tree Regressor weighs most in the final outcome. Table 4.4 shows the results for each of the transformations described in the previous section. The results are presented as the mean of cross-validation with standard deviation. The table shows that the ensemble model with logit transformation applied achieves a $3.82 \pm 0.18\%$ mean absolute error. It also turns out that looking at the regression analysis results in Table 4.2 and excluding all predictors that have nonzero p-values (except the method class variables MC2,MC3,MC4,MC5 and MC6), and could, therefore, be considered less statistically significant than other predictors, does not negatively impact the results. An ensemble regression model, with logit transformation and this reduced feature set, achieves a $3.99 \pm 0.2\%$ mean absolute error. If it does not perform significantly worse, a simpler model may be preferable, and therefore, we can choose to work with a reduced feature set. Figure 4.2 shows that the learning curve for this model starts to converge at 2000 training samples. The bias displayed between the training and validation curve is possibly due to noise in the data or other imperfections in the model. Figure 4.1 (b) shows the residuals for the ensemble regression model on overall accuracy. There is a significant improvement in heteroskedasticity compared to the least-squares model, and all predicted values fall within the bounds of our dependent variable.



(a) (b)

Figure 4.3: (a) Overall Sampling of Methods by Hyperspectral Image. (b) Distribution of overall accuracy by dataset.

4.5. Leave-One-Out Results

An experiment is also run on the prediction model, where we leave one of the hyperspectral images out of the training data and use it to test the performance of the model. Table 4.5 shows the results for this experiment. It shows a significantly worse performance, when leaving some images out of the training dataset.

Table 4.5: Leave-one-out prediction results.

Image	Mean absolute error [%]
Indian Pines	8.79
Pavia University	9.23
Pavia Centre	7.26
Salinas Valley	7.3
Salinas A-scene	5.96
Houston	10.6
Botswana	11.82
Washing DC Mall	6.13
Kennedy Space Center	6.34

5. Conclusion

In this study, an attempt has been made to create a model that can predict the performance of a hyperspectral image classification method from details of the method and the experiment used for the prediction. Features have been identified that describe how the proposed method and experimental setup can differ. Regression analysis gives some indication as to the deciding features by interpreting coefficients and p-values. An ordinary least squares regression model gives fairly decent results at 7.4% mean absolute error, beating a baseline model by over one percent.

Results show that having a nonlinear model is the deciding factor to improve performance. A nonlinear ensemble regressor model is chosen and optimized with Bayesian optimization. Results, 3.82% mean absolute error, show that the ensemble model can predict outcomes of methods to within 5%, based on attributes and experimental setup. Heteroskedasticity in residuals is looking much better and the learning curve suggests that more data does not yield better results.

Experiments, leaving one hyperspectral image out of the training dataset, suggest that the ensemble model does not perform well on a blind prediction with an unseen image. Despite some issues, the prediction model can provide a useful analysis tool for researchers to guide future research. Perhaps a tool such as this could be made part of online aggregators such as the IEEE library, where researchers could quickly cull together labeled data from hundreds of papers and make predictions and analyses to use in their own research.

A possible application of this work is in the domain of model optimization. Whereas current model optimization techniques, such as autosklearn, use Bayesian optimization or grid search methods to find an optimum classification model based on the input data, this prediction model could provide very useful information for model optimization with context from the problem domain. Suppose you want to create a classification map of a hyperspectral image, a rural area with crop fields, not unlike Indian Pines. A model search optimization would take the image itself as input and search the solution space for a good fitting model, which can be very computationally expensive. This model prediction optimization could take as input metadata about the image: resolution, number of classes, number of training samples, and so on and search the method categories for the optimal type of model based on thousands of experiments carried out in peer-reviewed research papers. As an example, you might think to create a 3D CNN model, but our model prediction optimization suggests that a 2D CNN model combined with an EMAP processor would yield the best solution. All that remains is to tune this particular model using traditional search optimization techniques. A model search optimization technique could never yield this type of insight.

There is one area of improvement for this study, in my opinion. When reviewing the method categories, it becomes evident that some of them constitute a preprocessing step or a feature extraction step while others are the classification algorithm itself. Therefore, the prediction model developed in this study does not support combining these steps into a pipeline. For example, when studying a classification method based on mathematical morphology, there is no distinction given whether the classification algorithm used is an SVM or a logistic regression. Changing the prediction model to include this information could possibly improve prediction results. In the hypothetical application of model optimization, this improved prediction model could return a complete processing pipeline instead of just the method category that could provide the best results.

Bibliography

[1] A.F.H. Goetz and G. Vane. Imaging spectrometry for earth remote sensing. *Science*, 228(4704), 1985.

[2] D. Haboudane, J.R. Miller, and I.B. Strachan E. Pattey, P.J. Zarco-Tejada. Hyperspectral vegetation indices and novel algorithms for predicting green lai of crop canopies: Modeling and validation in the context of precision agriculture. *Remote Sensing Of Environment*, 90(3):337–352.

[3] FA. Kruse. Mapping surface mineralogy using imaging spectrometry. *Geomorphology*, 137(1):41–56, 12.

[4] H. Ren and C.I. Chang. Automatic spectral target recognition in hyperspectral imagery. *IEEE Transactions on Aerospace and Electronic Systems*, 39(4):1232–1249, 2003.

[5] F. Melgani and L. Bruzzone. Classification of hyperspectral remote sensing images with support vector machines. 42(8):1778–1790, 2004.

[6] P. Ghamisi, J.A. Benediktsson, and M.O. Ulfarsson. Spectral–spatial classification of hyperspectral images based on hidden markov random fields. *IEEE Transactions on Geoscience and Remote Sensing*, 52(5):2565 – 2574, 2013.

[7] Y.S. Chen, ZH Z.H. Lin, X. Zhao, G. Wang, and Y.F. Gu. Deep learning-based classification of hyperspectral data. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 7(6):2094–2107, 2014.

[8] P Ghamisi, J. Plaza, Y. Chen, J. Li, and A. Plaza. Advanced supervised spectral classifiers for hyperspectral images: A review. *IEEE Geoscience and Remote Sensing Magazine (GRSM)*, 2017.

[9] P. Ghamisi, E. Maggiori, S. Li, R. Souza, Y. Tarabalka, G. Moser, A. De Giorgi, L. Fang, Y. Chen, M. Chi, S.B. Serpico, and J. A. Benediktsson. Frontiers in spectral-spatial classification of hyperspectral images. *IEEE Geoscience and Remote Sensing Magazine*, 2018.

[10] R. Khatami, G. Mountrakis, and S. Stehman. A meta-analysis of remote sensing research on supervised pixel-basedland-cover image classification processes:

General guidelines for practitioners and future research. *Remote Sensing of Environment*.

[11] G. G. Wilkinson. Results and implications of a study of fifteen years of satellite image classification experiments. *Results and Implications of a Study of Fifteen Years of Satellite Image Classification Experiments*, 43(3):433–440, 2005.

[12] José M. Bioucas-Dias, Antonio Plaza, Nicolas Dobigeon, Mario Parente, Qian Du, Paul Gader, and Jocelyn Chanussot. Hyperspectral unmixing overview: Geometrical, statistical, and sparse regression-based approaches. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 5(2):354–379, 2012.

[13] Wallace M. Porter and Harry T. Enmark. A System Overview Of The Airborne Visible/Infrared Imaging Spectrometer (Aviris). In Gregg Vane, editor, *Imaging Spectroscopy II*, volume 0834, pages 22 – 31. International Society for Optics and Photonics, SPIE, 1987.

[14] S. G. Ungar. Overview of the earth observing one (eo-1) mission. In *IEEE International Geoscience and Remote Sensing Symposium*, volume 1, pages 568–571 vol.1, 2002.

[15] Peter A. Mitchell. Hyperspectral digital imagery collection experiment (HYDICE). In Joan B. Lurie, James J. Pearson, and Eugenio Zilioli, editors, *Geographic Information Systems, Photogrammetry, and Geological/Geophysical Remote Sensing*, volume 2587, pages 70 – 95. International Society for Optics and Photonics, SPIE, 1995.

[16] P. Ghamisi, E. Maggiori, S. Li, R. Souza, Y. Tarablaka, G. Moser, A. De Giorgi, L. Fang, Y. Chen, M. Chi, S. B. Serpico, and J. A. Benediktsson. New frontiers in spectral-spatial hyperspectral image classification: The latest advances based on mathematical morphology, markov random fields, segmentation, sparse representation, and deep learning. *IEEE Geoscience and Remote Sensing Magazine*, 6(3):10–43, 2018.

[17] X. Yang, Y. Ye, X. Li, R. Y. K. Lau, X. Zhang, and X. Huang. Hyperspectral image classification with deep learning models. *IEEE Transactions on Geoscience and Remote Sensing*, 56(9):5408–5423, 2018.

[18] 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*, 57(9):6690–6709, 2019.

[19] M. Fauvel, Y. Tarabalka, J. A. Benediktsson, J. Chanussot, and J. C. Tilton. Advances in spectral-spatial classification of hyperspectral images. *Proceedings of the IEEE*, 101(3):652–675, 2013.

[20] H. Chen, F. Miao, and X. Shen. Hyperspectral remote sensing image classification with cnn based on quantum genetic-optimized sparse representation. *IEEE Access*, 8:99900–99909, 2020.

[21] Z. Xue, P. Du, and H. Su. Kernelized sparse graph-embedded dimensionality reduction for hyperspectral image classification. In *2014 6th Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS)*, pages 1–4, 2014.

[22] C. Cortes and V. Vapnik. Support vector networks. *Machine Learning*, 20:273–297, 1995.

[23] Tin Kam Ho. Random decision forests. In *Proceedings of 3rd International Conference on Document Analysis and Recognition*, volume 1, pages 278–282 vol.1, 1995.

[24] X. Zhang, Q. Song, Y. Zheng, B. Hou, and S. Gou. Classification of imbalanced hyperspectral imagery data using support vector sampling. In *2014 IEEE Geoscience and Remote Sensing Symposium*, pages 2870–2873, 2014.

[25] Kévin Bernard, Yuliya Tarabalka, Jesús Angulo, Jocelyn Chanussot, and Jón Atli Benediktsson. Spectral–spatial classification of hyperspectral data based on a stochastic minimum spanning forest approach. *IEEE Transactions on Image Processing*, 21(4):2008–2021, 2012.

[26] Xudong Kang, Shutao Li, Leyuan Fang, Meixiu Li, and Jón Atli Benediktsson. Extended random walker-based classification of hyperspectral images. *IEEE Transactions on Geoscience and Remote Sensing*, 53(1):144–153, 2015.

[27] X. Zhang, Y. sun, and W. Qi. Hyperspectral image classification based on extended morphological attribute profiles and abundance information. In *2018 9th Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS)*, pages 1–5, 2018.

[28] B. Song, J. Li, P. Li, and A. Plaza. Decision fusion based on extended multi-attribute profiles for hyperspectral image classification. In *2013 5th Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS)*, pages 1–4, 2013.

[29] X. Huang, X. Guan, J. A. Benediktsson, L. Zhang, J. Li, A. Plaza, and M. Dalla Mura. Multiple morphological profiles from multicomponent-base images for hyperspectral image classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 7(12):4653–4669, 2014.

[30] K. Wang, R. Huang, and Q. Song. Spectral-spatial hyperspectral image classification using extended multi attribute profiles and guided bilateral filter. In *2015 International Conference on Computer Science and Mechanical Automation (CSMA)*, pages 235–239, 2015.

[31] B. Song, J. Li, M. Dalla Mura, P. Li, A. Plaza, J. M. Bioucas-Dias, J. A. Benediktsson, and J. Chanussot. Remotely sensed image classification using sparse representations of morphological attribute profiles. *IEEE Transactions on Geoscience and Remote Sensing*, 52(8):5122–5136, 2014.

[32] M. Jouni, M. D. Mura, and P. Comon. Hyperspectral image classification using tensor cp decomposition. In *IGARSS 2019 - 2019 IEEE International Geoscience and Remote Sensing Symposium*, pages 1164–1167, 2019.

[33] J. A. Benediktsson, J. A. Palmason, and J. R. Sveinsson. Classification of hyperspectral data from urban areas based on extended morphological profiles. *IEEE Transactions on Geoscience and Remote Sensing*, 43(3):480–491, 2005.

[34] A. Plaza, P. Martinez, J. Plaza, and R. Perez. Dimensionality reduction and classification of hyperspectral image data using sequences of extended morphological transformations. *IEEE Transactions on Geoscience and Remote Sensing*, 43(3):466–479, 2005.

[35] A. Plaza and J. Plaza. Parallel morphological classification of hyperspectral imagery using extended opening and closing by reconstruction operations. In *IGARSS 2008 - 2008 IEEE International Geoscience and Remote Sensing Symposium*, volume 1, pages I–58–I–61, 2008.

[36] J. Liao and L. Wang. Adaptive hyperspectral image classification based on the fusion of manifolds filter and spatial correlation features. *IEEE Access*, 8:90390–90409, 2020.

[37] P. Sidike, C. Chen, V. Asari, Y. Xu, and W. Li. Classification of hyperspectral image using multiscale spatial texture features. In *2016 8th Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS)*, pages 1–4, 2016.

[38] C. Chen, J. Jiang, B. Zhang, W. Yang, and J. Guo. Hyperspectral image classification using gradient local auto-correlations. In *2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR)*, pages 454–458, 2015.

[39] M. Dalla Mura, A. Villa, J. A. Benediktsson, J. Chanussot, and L. Bruzzone. Classification of hyperspectral images by using extended morphological attribute profiles and independent component analysis. *IEEE Geoscience and Remote Sensing Letters*, 8(3):542–546, 2011.

[40] Prashanth R. Marpu, Mattia Pedergnana, Mauro Dalla Mura, Jon Atli Benediktsson, and Lorenzo Bruzzone. Automatic generation of standard deviation attribute profiles for spectral–spatial classification of remote sensing data. *IEEE Geoscience and Remote Sensing Letters*, 10(2):293–297, 2013.

[41] Pablo Quesada-Barriuso, Francisco Argüello, and Dora B. Heras. Spectral–spatial classification of hyperspectral images using wavelets and extended morphological profiles. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 7(4):1177–1185, 2014.

[42] B. Waske, S. van der Linden, J. A. Benediktsson, A. Rabe, and P. Hostert. Impact of different morphological profiles on the classification accuracy of urban hyperspectral data. In *2009 First Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing*, pages 1–4, 2009.

[43] Hamid Ghanbari, Saeid Homayouni, Abdolreza Safari, and Pedram Ghamisi. Gaussian mixture model and markov random fields for hyperspectral image classification. *European Journal of Remote Sensing*, 51(1):889–900, 2018.

[44] Y. Hu, S. T. Monteiro, and E. Saber. Comparing inference methods for conditional random fields for hyperspectral image classification. In *2015 7th Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS)*, pages 1–4, 2015.

[45] X. Cao, F. Zhou, L. Xu, D. Meng, Z. Xu, and J. Paisley. Hyperspectral image classification with markov random fields and a convolutional neural network. *IEEE Transactions on Image Processing*, 27(5):2354–2367, 2018.

[46] Y. Tarabalka, M. Fauvel, J. Chanussot, and J. A. Benediktsson. Svm- and mrf-based method for accurate classification of hyperspectral images. *IEEE Geoscience and Remote Sensing Letters*, 7(4):736–740, 2010.

[47] L. Sun, Z. Wu, J. Liu, L. Xiao, and Z. Wei. Supervised spectral–spatial hyperspectral image classification with weighted markov random fields. *IEEE Transactions on Geoscience and Remote Sensing*, 53(3):1490–1503, 2015.

[48] G. Moser and S. B. Serpico. Combining support vector machines and markov random fields in an integrated framework for contextual image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 51(5):2734–2752, 2013.

[49] P. Ghamisi, J. A. Benediktsson, and M. O. Ulfarsson. Spectral–spatial classification of hyperspectral images based on hidden markov random fields. *IEEE Transactions on Geoscience and Remote Sensing*, 52(5):2565–2574, 2014.

[50] Bor-Chen Kuo, Chun-Hsiang Chuang, Chih-Sheng Huang, and Chih-Cheng Hung. A nonparametric contextual classification based on markov random fields. In *2009 First Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing*, pages 1–4, 2009.

[51] J. Li, J. M. Bioucas-Dias, and A. Plaza. Spectral–spatial hyperspectral image segmentation using subspace multinomial logistic regression and markov random fields. *IEEE Transactions on Geoscience and Remote Sensing*, 50(3):809–823, 2012.

[52] B. Zhang, S. Li, X. Jia, L. Gao, and M. Peng. Adaptive markov random field approach for classification of hyperspectral imagery. *IEEE Geoscience and Remote Sensing Letters*, 8(5):973–977, 2011.

[53] Utsav B. Gewali and Sildomar T. Monteiro. Spectral angle based unary energy functions for spatial-spectral hyperspectral classification using markov random fields. In *2016 8th Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS)*, pages 1–6, 2016.

[54] Yang Xu, Zebin Wu, and Zhihui Wei. Markov random field with homogeneous areas priors for hyperspectral image classification. In *2014 IEEE Geoscience and Remote Sensing Symposium*, pages 3426–3429, 2014.

[55] Pedram Ghamisi, Jon Atli Benediktsson, and Magnus O. Ulfarsson. The spectral-spatial classification of hyperspectral images based on hidden markov random field and its expectation-maximization. In *2013 IEEE International Geoscience and Remote Sensing Symposium - IGARSS*, pages 1107–1110, 2013.

[56] Andong Ma and Anthony M. Filippi. Hyperspectral image classification via object-oriented segmentation-based sequential feature extraction and recurrent neural network. In *IGARSS 2020 - 2020 IEEE International Geoscience and Remote Sensing Symposium*, pages 72–75, 2020.

[57] Q. Jackson and D.A. Landgrebe. Adaptive bayesian contextual classification based on markov random fields. *IEEE Transactions on Geoscience and Remote Sensing*, 40(11):2454–2463, 2002.

[58] Yuan Yuan, Jianzhe Lin, and Qi Wang. Hyperspectral image classification via multitask joint sparse representation and stepwise mrf optimization. *IEEE Transactions on Cybernetics*, 46(12):2966–2977, 2016.

[59] Jun Li, José M. Bioucas-Dias, and Antonio Plaza. Spectral–spatial classification of hyperspectral data using loopy belief propagation and active learning. *IEEE Transactions on Geoscience and Remote Sensing*, 51(2):844–856, 2013.

[60] X. Kang, S. Li, and J. A. Benediktsson. Spectral–spatial hyperspectral image classification with edge-preserving filtering. *IEEE Transactions on Geoscience and Remote Sensing*, 52(5):2666–2677, 2014.

[61] Y. Tarabalka, J. A. Benediktsson, J. Chanussot, and J. C. Tilton. Multiple spectral–spatial classification approach for hyperspectral data. *IEEE Transactions on Geoscience and Remote Sensing*, 48(11):4122–4132, 2010.

[62] Davood Akbari, Saeid Homayouni, Abdolreza Safari, Safa Khazai, and Hossein Torabzadeh. An improved marker selection method for hyperspectral image segmentation and classification. In *2014 6th Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS)*, pages 1–4, 2014.

[63] Leyuan Fang, Shutao Li, Wuhui Duan, Jinchang Ren, and Jón Atli Benediktsson. Classification of hyperspectral images by exploiting spectral–spatial information of superpixel via multiple kernels. *IEEE Transactions on Geoscience and Remote Sensing*, 53(12):6663–6674, 2015.

[64] Y. Tarabalka, J. A. Benediktsson, and J. Chanussot. Spectral–spatial classification of hyperspectral imagery based on partitional clustering techniques. *IEEE Transactions on Geoscience and Remote Sensing*, 47(8):2973–2987, 2009.

[65] P. Ghamisi, M. S. Couceiro, F. M. L. Martins, and J. A. Benediktsson. Multilevel image segmentation based on fractional-order darwinian particle swarm optimization. *IEEE Transactions on Geoscience and Remote Sensing*, 52(5):2382–2394, 2014.

[66] Yuliya Tarabalka and James C. Tilton. Spectral-spatial classification of hyperspectral images using hierarchical optimization. In *2011 3rd Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS)*, pages 1–4, 2011.

[67] X. Zhang, P. Weng, J. Feng, E. Zhang, and B. Hou. Spatial-spectral classification based on group sparse coding for hyperspectral image. In *2013 IEEE International Geoscience and Remote Sensing Symposium - IGARSS*, pages 1745–1748, 2013.

[68] M. Cui and S. Prasad. Multiscale sparse representation classification for robust hyperspectral image analysis. In *2013 IEEE Global Conference on Signal and Information Processing*, pages 969–972, 2013.

[69] A. Sumarsono and Q. Du. Hyperspectral image classification with low-rank subspace and sparse representation. In *2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, pages 2864–2867, 2015.

[70] Y. Chen, N. M. Nasrabadi, and T. D. Tran. Hyperspectral image classification using dictionary-based sparse representation. *IEEE Transactions on Geoscience and Remote Sensing*, 49(10):3973–3985, 2011.

[71] Y. Chen, N. M. Nasrabadi, and T. D. Tran. Hyperspectral image classification via kernel sparse representation. *IEEE Transactions on Geoscience and Remote Sensing*, 51(1):217–231, 2013.

[72] Y. Qian, M. Ye, and J. Zhou. Hyperspectral image classification based on structured sparse logistic regression and three-dimensional wavelet texture features. *IEEE Transactions on Geoscience and Remote Sensing*, 51(4):2276–2291, 2013.

[73] L. Fang, S. Li, X. Kang, and J. A. Benediktsson. Spectral–spatial hyperspectral image classification via multiscale adaptive sparse representation. *IEEE Transactions on Geoscience and Remote Sensing*, 52(12):7738–7749, 2014.

[74] H. Zhang, J. Li, Y. Huang, and L. Zhang. A nonlocal weighted joint sparse representation classification method for hyperspectral imagery. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 7(6):2056–2065, 2014.

[75] Haoyang Yu, Xiao Zhang, Meiping Song, Jiaochan Hu, and Lianru Gao. Superpixel-level constraint representation for hyperspectral imagery classification. In *IGARSS 2020 - 2020 IEEE International Geoscience and Remote Sensing Symposium*, pages 56–59, 2020.

[76] Leyuan Fang, Shutao Li, Xudong Kang, and Jón Atli Benediktsson. Spectral–spatial classification of hyperspectral images with a superpixel-based discriminative sparse model. *IEEE Transactions on Geoscience and Remote Sensing*, 53(8):4186–4201, 2015.

[77] Warren Mcculloch and Walter Pitts. A logical calculus of ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, 5:127–147, 1943.

[78] L. Mou, P. Ghamisi, and X. X. Zhu. Deep recurrent neural networks for hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 55(7):3639–3655, 2017.

[79] Xiaorui Ma, Hongyu Wang, and Jie Geng. Spectral–spatial classification of hyperspectral image based on deep auto-encoder. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 9(9):4073–4085, 2016.

[80] L. Fang, G. Liu, S. Li, P. Ghamisi, and J. A. Benediktsson. Hyperspectral image classification with squeeze multibias network. *IEEE Transactions on Geoscience and Remote Sensing*, 57(3):1291–1301, 2019.

[81] Guangzhe Zhao, Guangyun Liu, Leyuan Fang, Bing Tu, and Pedram Ghamisi. Multiple convolutional layers fusion framework for hyperspectral image classification. *Neurocomputing*, 339, 02 2019.

[82] Y. Chen, K. Zhu, L. Zhu, X. He, P. Ghamisi, and J. A. Benediktsson. Automatic design of convolutional neural network for hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 57(9):7048–7066, 2019.

[83] S. K. Roy, G. Krishna, S. R. Dubey, and B. B. Chaudhuri. Hybridsn: Exploring 3-d–2-d cnn feature hierarchy for hyperspectral image classification. *IEEE Geoscience and Remote Sensing Letters*, 17(2):277–281, 2020.

[84] 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*, 55(9):5293–5301, 2017.

[85] R. S. W. Chu, H. Ng, X. Wang, and W. Luk. Convolution based spectral partitioning architecture for hyperspectral image classification. In *IGARSS 2019 - 2019 IEEE International Geoscience and Remote Sensing Symposium*, pages 3962–3965, 2019.

[86] M. E. Paoletti, J. M. Haut, R. Fernandez-Beltran, J. Plaza, A. J. Plaza, and F. Pla. Deep pyramidal residual networks for spectral–spatial hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 57(2):740–754, 2019.

[87] Wenju Wang, Shuguang Dou, Zhongmin Jiang, and Liujie Sun. A fast dense spectral–spatial convolution network framework for hyperspectral images classification. *Remote Sensing*, 10:1068, 07 2018.

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

[89] S. K. Roy, S. R. Dubey, S. Chatterjee, and B. Baran Chaudhuri. Fusenet: fused squeeze-and-excitation network for spectral-spatial hyperspectral image classification. *IET Image Processing*, 14(8):1653–1661, 2020.

[90] J. Zheng, Y. Feng, C. Bai, and J. Zhang. Hyperspectral image classification using mixed convolutions and covariance pooling. *IEEE Transactions on Geoscience and Remote Sensing*, pages 1–13, 2020.

[91] M. Zhu, L. Jiao, F. Liu, S. Yang, and J. Wang. Residual spectral-spatial attention network for hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing*, pages 1–14, 2020.

[92] H. Huang, C. Pu, Y. Li, and Y. Duan. Adaptive residual convolutional neural network for hyperspectral image classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 13:2520–2531, 2020.

[93] Sheng Wan, Chen Gong, Ping Zhong, Shirui Pan, Guangyu Li, and Jian Yang. Hyperspectral image classification with context-aware dynamic graph convolutional network. *IEEE Transactions on Geoscience and Remote Sensing*, PP:1–16, 05 2020.

[94] Z. Zhong, J. Li, L. Ma, H. Jiang, and H. Zhao. Deep residual networks for hyperspectral image classification. In *2017 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, pages 1824–1827, 2017.

[95] Y. Chen, H. Jiang, C. Li, X. Jia, and P. Ghamisi. Deep feature extraction and classification of hyperspectral images based on convolutional neural networks. *IEEE Transactions on Geoscience and Remote Sensing*, 54(10):6232–6251, 2016.

[96] Indira Bidari, Satyadhyan Chickerur, Harivijay Ranmale, Sushmita Talawar, Harish Ramadurg, and Rekha Talikoti. Hyperspectral imagery classification using deep learning. In *2020 Fourth World Conference on Smart Trends in Systems, Security and Sustainability (WorldS4)*, pages 672–676, 2020.

[97] Zhengying Li, Hong Huang, and Chunyu Pu. Deep manifold learning network for hyperspectral image classification. In *IGARSS 2020 - 2020 IEEE International Geoscience and Remote Sensing Symposium*, pages 2021–2024, 2020.

[98] Yu Li, Lei Zhang, Wei Wei, and Yanning Zhang. Deep self-supervised learning for few-shot hyperspectral image classification. In *IGARSS 2020 - 2020 IEEE International Geoscience and Remote Sensing Symposium*, pages 501–504, 2020.

[99] Xin He, Yushi Chen, and Pedram Ghamisi. Dual graph convolutional network for hyperspectral image classification with limited training samples. *IEEE Transactions on Geoscience and Remote Sensing*, pages 1–18, 2021.

[100] Erchan Aptoula, Murat Can Ozdemir, and Berrin Yanikoglu. Deep learning with attribute profiles for hyperspectral image classification. *IEEE Geoscience and Remote Sensing Letters*, 13(12):1970–1974, 2016.

[101] Wenzhi Zhao and Shihong Du. Spectral–spatial feature extraction for hyperspectral image classification: A dimension reduction and deep learning approach. *IEEE Transactions on Geoscience and Remote Sensing*, 54(8):4544–4554, 2016.

[102] Wei Li, Guodong Wu, Fan Zhang, and Qian Du. Hyperspectral image classification using deep pixel-pair features. *IEEE Transactions on Geoscience and Remote Sensing*, 55(2):844–853, 2017.

[103] Zilong Zhong, Jonathan Li, Zhiming Luo, and Michael Chapman. Spectral–spatial residual network for hyperspectral image classification: A 3-d deep learning framework. *IEEE Transactions on Geoscience and Remote Sensing*, 56(2):847–858, 2018.

[104] Bin Pan, Zhenwei Shi, and Xia Xu. R-vcanet: A new deep-learning-based hyperspectral image classification method. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 10(5):1975–1986, 2017.

[105] Weiwei Song, Shutao Li, Leyuan Fang, and Ting Lu. Hyperspectral image classification with deep feature fusion network. *IEEE Transactions on Geoscience and Remote Sensing*, 56(6):3173–3184, 2018.

[106] Pedram Ghamisi, Yushi Chen, and Xiao Xiang Zhu. A self-improving convolution neural network for the classification of hyperspectral data. *IEEE Geoscience and Remote Sensing Letters*, 13(10):1537–1541, 2016.

[107] Lin Zhu, Yushi Chen, Pedram Ghamisi, and Jón Atli Benediktsson. Generative adversarial networks for hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 56(9):5046–5063, 2018.

[108] Jingxiang Yang, Yong-Qiang Zhao, and Jonathan Cheung-Wai Chan. Learning and transferring deep joint spectral–spatial features for hyperspectral classification. *IEEE Transactions on Geoscience and Remote Sensing*, 55(8):4729–4742, 2017.

[109] Hao Wu and Saurabh Prasad. Semi-supervised deep learning using pseudo labels for hyperspectral image classification. *IEEE Transactions on Image Processing*, 27(3):1259–1270, 2018.

[110] G. E. Hinton, S. Osindero, and Y. W. Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18:1527–1554, 2006.

[111] Yushi Chen, Xing Zhao, and Xiuping Jia. Spectral–spatial classification of hyperspectral data based on deep belief network. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 8(6):2381–2392, 2015.

[112] Ping Zhong, Zhiqiang Gong, Shutao Li, and Carola-Bibiane Schönlieb. Learning to diversify deep belief networks for hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 55(6):3516–3530, 2017.

[113] Andrey N. Tikhonov and Vasiliy Y. Arsenin. *Solutions of ill-posed problems*. V. H. Winston & Sons, Washington, D.C.: John Wiley & Sons, New York, 1977. Translated from the Russian, Preface by translation editor Fritz John, Scripta Series in Mathematics.

[114] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society (Series B)*, 58:267–288, 1996.

[115] Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society, Series B*, 67:301–320, 2005.

[116] S.G. Mallat and Zhifeng Zhang. Matching pursuits with time-frequency dictionaries. *IEEE Transactions on Signal Processing*, 41(12):3397–3415, 1993.

[117] Harris Drucker, Chris, Burges* L. Kaufman, Alex Smola, and Vladimir Vapnik. Support vector regression machines. In *Advances in Neural Information Processing Systems 9*, volume 9, pages 155–161, 1997.

[118] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Wadsworth and Brooks, Monterey, CA, 1984.

[119] Y. Freund and R. E. Schapire. A Decision Theoretic Generalization of On-Line Learning and an Application to Boosting. In Paul M. B. Vitányi, editor, *Second European Conference on Computational Learning Theory (EuroCOLT-95)*, pages 23–37, 1995.

[120] Aleksei Guryanov. Histogram-based algorithm for building gradient boosting ensembles of piecewise linear decision trees. In Wil M. P. van der Aalst, Vladimir Batagelj, Dmitry I. Ignatov, Michael Khachay, Valentina Kuskova, Andrey Kutuzov, Sergei O. Kuznetsov, Irina A. Lomazova, Natalia Loukachevitch, Amedeo Napoli, Panos M. Pardalos, Marcello Pelillo, Andrey V. Savchenko, and Elena Tutubalina, editors, *Analysis of Images, Social Networks and Texts*, pages 39–50, Cham, 2019. Springer International Publishing.

A. Average Accuracy Results

Table A.1: Average Accuracy prediction results for linear regression models. Values are given as mean absolute error [%] with standard deviation.

	Transformations					
	No	Logit	Standard	Power	Gaussian	Uniform
Baseline	8.4 ± 0.64	7.98 ± 0.88	8.4 ± 0.64	7.99 ± 0.81	7.95 ± 0.85	7.96 ± 0.85
Linear	7.37 ± 0.48	6.82 ± 0.6	7.37 ± 0.48	6.87 ± 0.62	6.79 ± 0.57	6.88 ± 0.67
Ridge	7.36 ± 0.49	6.82 ± 0.61	7.36 ± 0.49	6.88 ± 0.63	6.8 ± 0.59	6.9 ± 0.68
Lasso	7.67 ± 0.59	7.31 ± 0.77	7.67 ± 0.59	7.39 ± 0.73	7.26 ± 0.74	7.4 ± 0.77
OMP	7.79 ± 0.51	7.07 ± 0.63	7.79 ± 0.51	7.17 ± 0.65	7.07 ± 0.62	7.17 ± 0.7
ElasticNet	7.68 ± 0.6	7.34 ± 0.78	7.73 ± 0.6	7.46 ± 0.74	7.31 ± 0.75	7.43 ± 0.77
Average	$\mathbf{7.71 \pm 0.55}$	$\mathbf{7.22 \pm 0.71}$	$\mathbf{7.72 \pm 0.55}$	$\mathbf{7.29 \pm 0.7}$	$\mathbf{7.2 \pm 0.69}$	$\mathbf{7.29 \pm 0.74}$

Table A.2: Average accuracy prediction results for nonlinear regression models. Values are given as mean absolute error [%] with standard deviation.

	Transformations					
	No	Logit	Standard	Power	Gaussian	Uniform
Ensemble	4.07 ± 0.4	3.94 ± 0.39	4.07 ± 0.4	3.9 ± 0.38	3.94 ± 0.39	3.89 ± 0.38
SVR	4.9 ± 0.38	4.66 ± 0.33	4.8 ± 0.34	N/A	4.65 ± 0.32	4.79 ± 0.19
MLP	4.63 ± 0.36	4.57 ± 0.3	4.52 ± 0.33	N/A	4.4 ± 0.39	4.87 ± 0.36
Random forest	4.29 ± 0.41	4.12 ± 0.46	4.29 ± 0.41	4.09 ± 0.44	4.1 ± 0.48	4.1 ± 0.41
Ada boost	4.44 ± 0.42	4.35 ± 0.32	4.44 ± 0.42	N/A	4.25 ± 0.38	4.17 ± 0.39
Histogram gradient boosting	4.22 ± 0.4	4.14 ± 0.39	4.22 ± 0.4	N/A	4.13 ± 0.42	4.36 ± 0.4
KNN	4.25 ± 0.34	4.17 ± 0.33	4.25 ± 0.34	N/A	4.13 ± 0.35	4.05 ± 0.33
Average	$\mathbf{4.4 \pm 0.39}$	$\mathbf{4.28 \pm 0.36}$	$\mathbf{4.37 \pm 0.38}$	**N/A**	$\mathbf{4.23 \pm 0.39}$	$\mathbf{4.32 \pm 0.35}$

B. Cohen's Kappa Results

Table B.1: Cohen's Kappa prediction results for linear regression models. Values are given as mean absolute error ×100 with standard deviation.

	Transformations					
	No	Logit	Standard	Power	Gaussian	Uniform
Baseline	10.33 ± 0.55	9.87 ± 0.34	10.33 ± 0.55	9.86 ± 0.61	9.83 ± 0.64	9.83 ± 0.64
Linear	9.22 ± 0.42	8.58 ± 0.51	9.22 ± 0.42	8.69 ± 0.43	8.54 ± 0.47	8.62 ± 0.48
Ridge	9.22 ± 0.42	8.59 ± 0.51	9.23 ± 0.42	8.7 ± 0.44	8.55 ± 0.48	8.64 ± 0.48
Lasso	9.8 ± 0.5	9.09 ± 0.6	9.8 ± 0.5	9.25 ± 0.57	9.09 ± 0.58	9.26 ± 0.61
OMP	9.44 ± 0.43	8.9 ± 0.55	9.44 ± 0.43	8.97 ± 0.52	8.86 ± 0.52	8.93 ± 0.56
ElasticNet	9.81 ± 0.49	9.14 ± 0.61	9.83 ± 0.5	9.32 ± 0.58	9.13 ± 0.59	9.28 ± 0.61
Average	$\mathbf{9.64 \pm 0.47}$	$\mathbf{9.03 \pm 0.52}$	$\mathbf{9.64 \pm 0.47}$	$\mathbf{9.13 \pm 0.53}$	$\mathbf{9.0 \pm 0.55}$	$\mathbf{9.09 \pm 0.56}$

Table B.2: Cohen's Kappa prediction results for nonlinear regression models. Values are given as mean absolute error ×100 with standard deviation.

	Transformations					
	No	Logit	Standard	Power	Gaussian	Uniform
Ensemble	4.56 ± 0.15	4.45 ± 0.2	4.55 ± 0.16	4.46 ± 0.16	4.49 ± 0.17	4.47 ± 0.15
SVR	6.09 ± 0.24	5.83 ± 0.23	5.91 ± 0.2	N/A	5.81 ± 0.24	5.8 ± 0.24
MLP	5.19 ± 0.24	5.07 ± 0.27	5.08 ± 0.21	N/A	5.14 ± 0.3	5.34 ± 0.38
Random forest	4.62 ± 0.09	4.46 ± 0.15	4.62 ± 0.11	4.47 ± 0.11	4.5 ± 0.1	4.49 ± 0.11
Ada boost	4.68 ± 0.2	4.74 ± 0.15	4.68 ± 0.15	N/A	4.74 ± 0.19	4.71 ± 0.07
Histogram gradient boosting	4.68 ± 0.19	4.66 ± 0.19	4.68 ± 0.19	N/A	4.62 ± 0.18	4.86 ± 0.14
KNN	4.76 ± 0.19	4.68 ± 0.19	4.76 ± 0.19	N/A	4.71 ± 0.17	4.66 ± 0.16
Average	$\mathbf{4.94 \pm 0.18}$	$\mathbf{4.84 \pm 0.2}$	$\mathbf{4.6 \pm 0.17}$	**N/A**	$\mathbf{4.86 \pm 0.19}$	$\mathbf{4.9 \pm 0.18}$