# A Comprehensive Study of Unsupervised Classification Techniques for Hyperspectral Datasets

Himanshi Yadav

CMU-RI-TR-20-20
April 2020

The Robotics Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA

**Thesis Committee:**
David Wettergreen, *chair*
Ioannis Gkioulekas
Alberto Candela

*Submitted in partial fulfillment of the requirements for the degree of Master of Science in Robotics.*

*To everyone.*

# Abstract

Unsupervised learning and, in this specific research, clustering regional composition in hyperspectral images, poses significant challenges in the fields of machine learning and remote sensing. Hyperspectral images capture the spectral information in many wavelengths, as opposed to typical images that only capture three: red, blue and green. They are high-dimensional and have considerable noise and class overlap which add to the difficulty in experimentation, analysis and interpretation. This research conducts a comprehensive study of clustering techniques when applied to hyperspectral images. We focus on finding answers to some of the open questions present in the literature. We look at clustering techniques in terms of theoretical, algorithmic, and empirical differences. We evaluate the impact of changes in spectral and spatial resolution, and the number of classes present in the data. We also perform hyperparameter analysis of dimensionality reduction techniques. We observe that spatial information along with spectral information is important for clustering. It is also imperative to note that no one algorithm is applicable to all datasets and this remains an open question.

# Acknowledgments

Firstly, I would like to thank my advisor Prof. David Wettergreen for his support and guidance throughout my masters. His willingness and enthusiasm to help me with this research work has helped me considerably.

Then I would like to help my lab mates, Kevin Edelson, Suhit Kodgule, Srinivasan Vijayarangan and Alberto Candela for brainstorming ideas with me and providing constructive criticism.

I would like to thank everyone at Carnegie Mellon University for believing in me and providing their help wherever needed. In particular, my sincere thanks extend to Alison Lusk, John Dolan, Andrew Li, Ann Trevellini, Jean Harpley, BJ Fecich, Alison Day and Catherine Getchell.

Lastly, I would like to thank my support system. My parents have provided me with constant moral support. I owe a great debt of gratitude to my family and cousins who have been with me throughout this process. In Pittsburgh, I have found a home. I thank my dear friends Keene Chin, Gail Jones, Jane Hadburg and Anthony Harris. I also thank all my friends in Pittsburgh and back in India.

I also received considerable support from the research community. I would like to thank James Murphy for his support and providing several useful insights for this work. I would also like to thank Lloyd Windrim and Xifeng Guo for making their work open source and easy to reproduce.

This work was supported by the National Science Foundation National Robotics Initiative Grant #IIS-1526667. I gratefully acknowledge the assistance of Dr. David Thompson at Jet Propulsion Laboratory, California Institute of Technology.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

Unsupervised classification is a sub-category of machine learning algorithms which aims at classifying data into similar groups well distinguished from other groups. This type of classification inspects a dataset to then divide into possible groups or clusters based on commonalities. It does so without using any training data. Unsupervised classification is often referred to as clustering.

Clustering in itself is an ill-posed problem and requires several assumptions to be imposed on the data and measurement method to make it tractable [73]. These assumptions may be analytic, topological, statistical or geometric. In fact, there is no universal definition of a cluster and the methods depend on the data and the application.

## 1.1 Motivation

Imaging spectroscopy is the acquisition of two-dimensional images, using a spectrometer (Figure 1.1). Each pixel in the images contains information about energy in wavelengths of the electromagnetic spectrum. Instead of just visible light wavelengths of red, blue and green, a wide range of measurements are observed. In the past, the imaging spectroscopy community employed exhaustive libraries of identified materials to study new sites and objects. After collecting a spectrum, scientists compared it with spectral signatures present in the library to determine its physical and chemical properties [21]. However, the use of an exhaustive library creates several potential difficulties when the objective is to find new materials or mixtures. This is most significantly seen in the field of planetary exploration where new materials or unique mixtures are to be discovered at little-known or obscure places (Figure 1.1). We also see this in the fields of biology and geology. Spectral libraries usually contain pure minerals and samples but this is rarely the case in nature. There is a mix of geology or biology in a scene even at the pixel level. As a result there are mixing ratios and noise in different forms of life on Earth as well. Therefore, we can utilize unsupervised classification to solve this problem as it can successfully cluster a dataset into similar groups and the representative spectral signature of each group in itself leads to the discovery of a new material. Therefore, unsupervised classification can provide chief insights in unexplored regions and environments.

There are very few datasets present for experimentation. Datasets also suffer from a lack of sufficient labelled data. The objective of this work is to analyse and study unsupervised machine learning techniques that do not require labelled training data. We look at clustering techniques on certain low-resolution and high-resolution images.

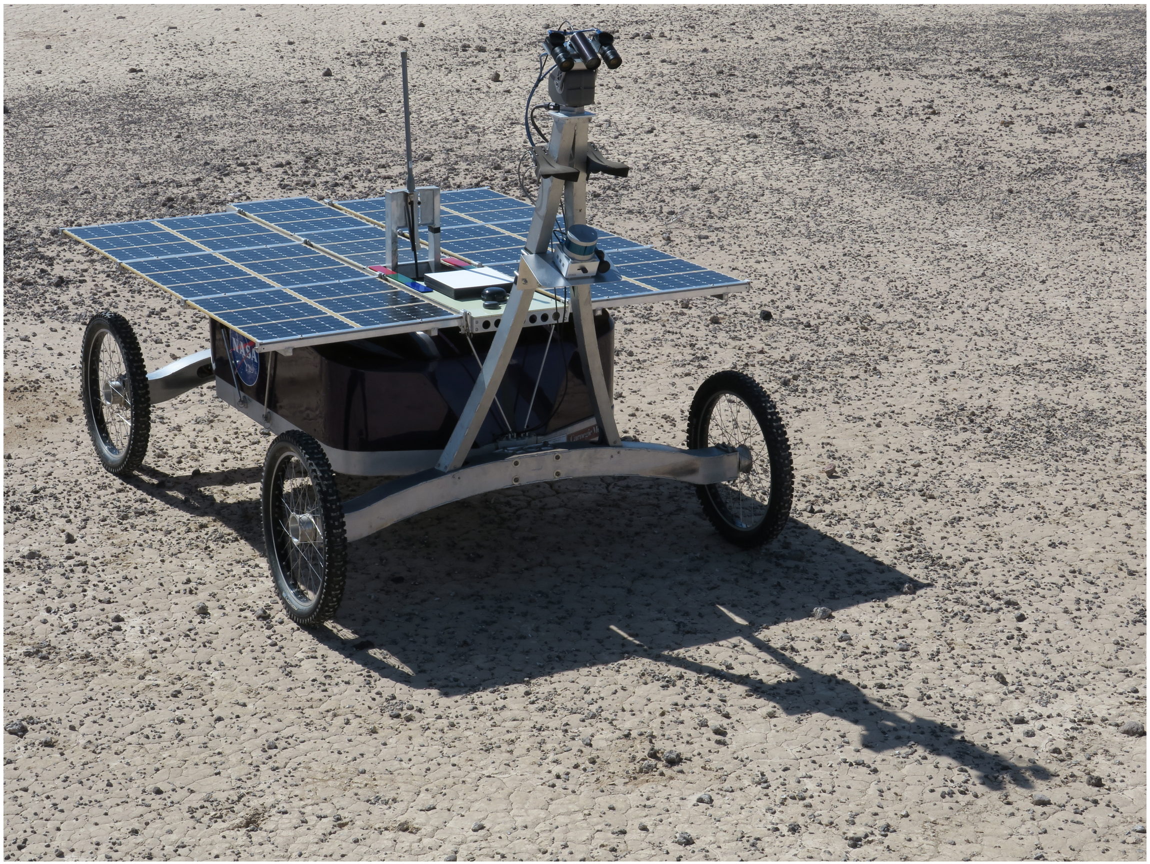


Figure 1.1: Zoë rover developed at Field Robotics Center, Robotics Institute, Carnegie Mellon University for planetary exploration, is equipped with an imaging spectrometer. The foreoptic (center) present on the mast and the white reference is present on the rover chassis.

## 1.2 Contributions

The contributions of this thesis to the fields of remote sensing, robotics and machine learning are as follows:

- We apply existing unsupervised machine learning techniques to low-resolution and high-resolution hyperspectral datasets, especially state-of-the-art techniques already in use for RGB datasets.
- We provide a thorough theoretical comparison of existing unsupervised clustering techniques.
- We study the effect of change in hyperspectral dataset size in terms of number of clusters, spatial and spectral resolution on existing unsupervised machine learning techniques.

- We provide empirical analysis of unsupervised dimensionality reduction techniques when applied to hyperspectral datasets.
- We study the impact of including or excluding unknown class data to unsupervised machine learning in the context of hyperspectral datasets.

## 1.3 Overview of the thesis

In this work, Chapter 2 introduces the concept of hyperspectral images and discusses in depth the various hyperspectral datasets used for experimentation. Chapter 3 discusses the related work in the area of unsupervised machine learning techniques. We describe the literature for dimensionality reduction and clustering hyperspectral datasets. In Chapter 4, we explain and overview the methods and algorithms used in this work. Chapter 5 looks at the theoretical comparative analysis of the techniques discussed in Chapter 4. Chapter 6 studies the approaches used in unsupervised classification empirically and we look at the major differences. Chapter 7 concludes this thesis and draws attention to the essential concepts discussed. We also summarize the major contributions of our work and delve into open problems to consider and potential next steps in our research.

# Chapter 2

# Hyperspectral Images

A hyperspectral image (HSI) is a two-dimensional spatial image with a spectral dimension to create a three-dimensional data cube as shown in Figure 2.1. Each pixel encodes a spectrum, as depicted in Figure 2.2, is primarily the measured response of the interaction between light and an object at a particular wavelength of light. There are several instruments that are used to acquire these hyperspectral images such as NASA AVIRIS-NG (Airborne Visible/Infrared Imaging Spectrometer- Next Generation) [54], AIS (Airborne Imaging Spectrometer), CASI (Compact Airborne Spectrographic Imager), and PROBE-1 [106]. The AVIRIS instrument was designed to measure the entire solar reflected spectrum from 400 to 2500 nanometers and the obtained image has 224 contiguous spectral bands.

In a hyperspectral image, the first two dimensions represent the spatial image and each pixel in the spatial image contains spectral information which is the third dimension to consider. The spectral information primarily stores molecular or composition information about an object whereas the spatial information helps with localization and positioning.

Each pixel in a hyperspectral image can be used to uniquely identify its content and can help differentiate a particular pixel from the rest. For example, a leaf would have a contrasting spectral signature to that of water as is depicted in Figure 2.3.

Remote sensing is the study and science of physical characteristics of a particular region to identify, measure, monitor and infer about specific objects without coming in physical contact with these objects [57]. It employs the electromagnetic wave emitted,

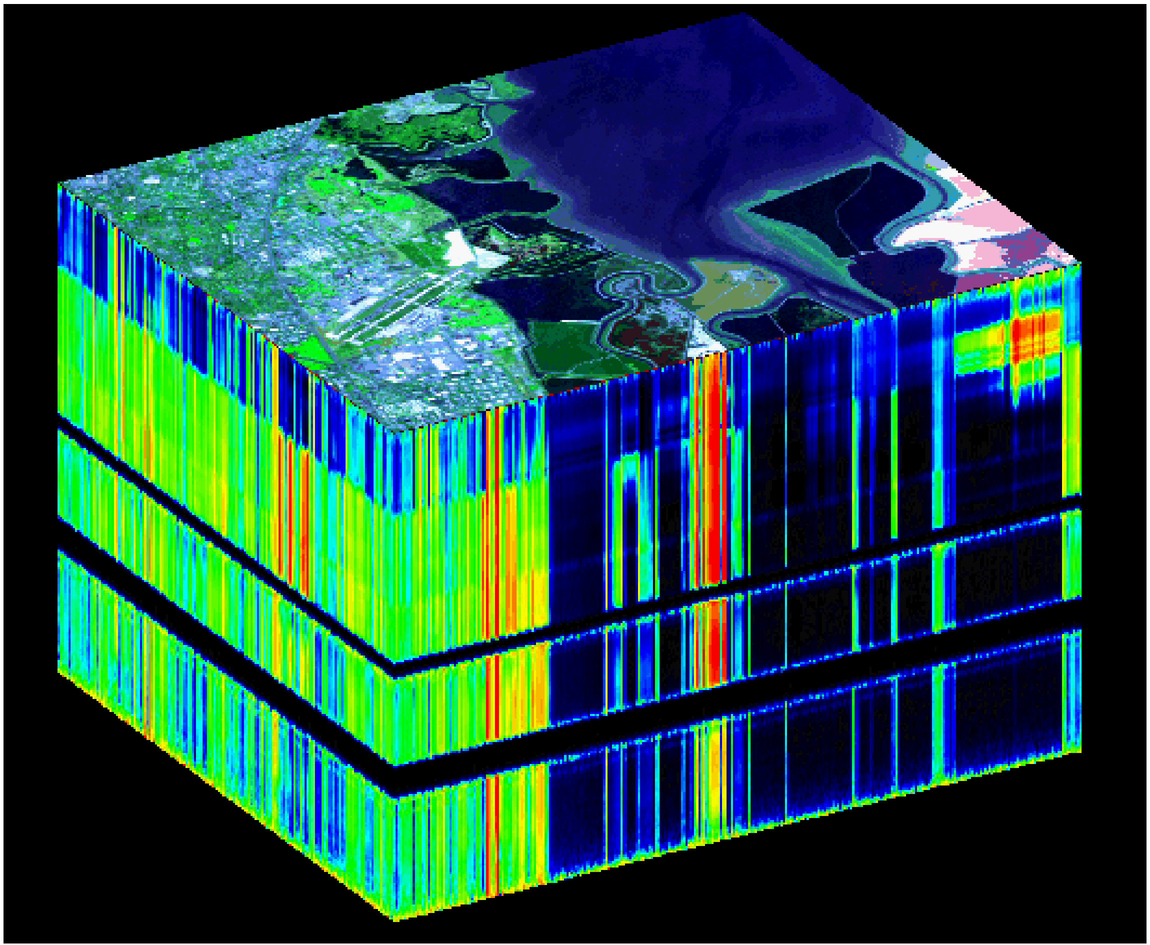


Figure 2.1: A hyperspectral image. The image was obtained from JPL's Airborne Visible/Infrared Imaging Spectrometer (AVIRIS) on August 20, 1992. The instrument was mounted on a NASA ER-2 plane and the image captured at an altitude of 20,000 meters (65,000 feet) over Moffett Field, California, at the southern end of the San Francisco Bay.

reflected and diffracted by the targeted area to quantify the physical characteristics of the area. Modern remote sensing is not limited to visible light and incorporates the entire electromagnetic spectrum spanning from ultraviolet waves to infrared, and microwaves. The third dimension of a hyperspectral image holds the spectral information which is obtained by measuring radiation and storing in the form of a spectral band. A spectral band is obtained by discretizing the electromagnetic spectrum.

In remote sensing, there has been a shift from multi-spectral imaging to hyperspectral imagining. Multi-spectral imagery has fewer number of bands compared to hyperspectral imagery. The bands in a hyperspectral image are narrower, discon-

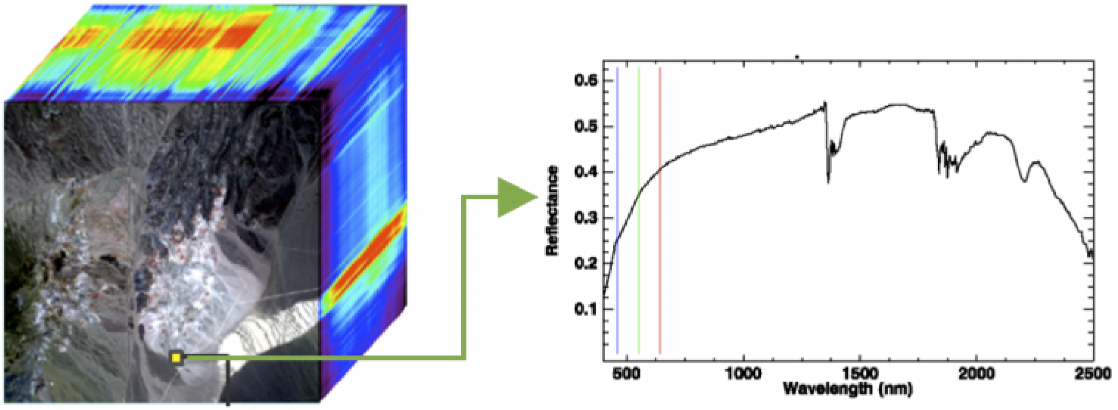


Figure 2.2: On the left is an example of a hyperspectral image of Cuprite, Nevada, USA. On the right, a single pixel in the hyperspectral image is zoomed in and visualized. There are more than the three RGB bands (can be seen on the far left of the graph) in this spectrum. [47]

tinuous and wider, and often are selected to be diagnostic of a particular property. Multi-spectral imaging uses a remote sensing radiometer to obtain the image; hyperspectral images are obtained using an imaging spectrometer. For example, the LANDSAT [3] is a multi-spectral sensor. AVIRIS [2] is used for hyperspectral imagery. Hyperspectral imaging has the ability to obtain higher spectral resolution images which can help distinguish between materials better than multi-spectral imaging because it has more information.

Clustering of hyperspectral images is a challenging task due to the large data size. With the advancement in imaging spectroscopy and measurement devices, we now have hyperspectral images with a higher spectral and spatial resolution [53]. These high-resolution high-dimensional hyperspectral images necessitate computationally faster and memory efficient algorithms. There is also a lack of generalizable algorithms that deal with the inherent noise and non-linearities present in hyperspectral datasets.

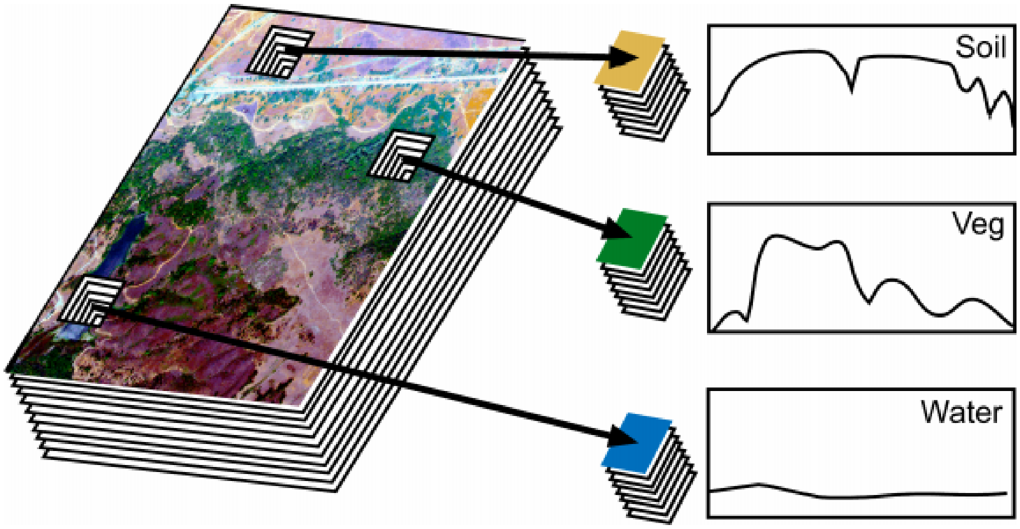


Figure 2.3: An imaging spectrometer records a spectrum for every pixel. Each pixel in turn can uniquely identify an object, in this case, we see water, soil and vegetation [97].

## 2.1 Applications of Hyperspectral Images

Hyperspectral images have numerous applications and an important sector where these images are used is agriculture. Visual examination of crops is limited by the discriminatory power of the human eye. A specific condition needs to be well-developed before experienced observers can detect it. Hyperspectral images can provide valuable information which greatly aids precision agriculture such as plant hydration and nutrition. Precision agriculture aims at spot application of advanced agricultural tactics instead of the whole field. The main advantages of hyperspectral imaging in precision agriculture are: low cost, consistent results, simplicity in usage, fast assessment, non-destructive, and highly accurate [26]. Jay et al. employ hyperspectral image based system to detect invasive weed infestation. Kanning et al. use UAV-based hyperspectral imagery to compute the yield of wheat as a function of the fertilizer concentration as shown in Figure 2.4. Other applications of hyperspectral images in agriculture include drought stress estimation [24], plant disease detection [44, 74], soil property analysis [13, 40, 49], and nutrient stress estimation [35, 85]. For example, the spectra from a hyperspectral image allows us to distinguish plant disease by detecting changes in leaf spectra over a period of time.

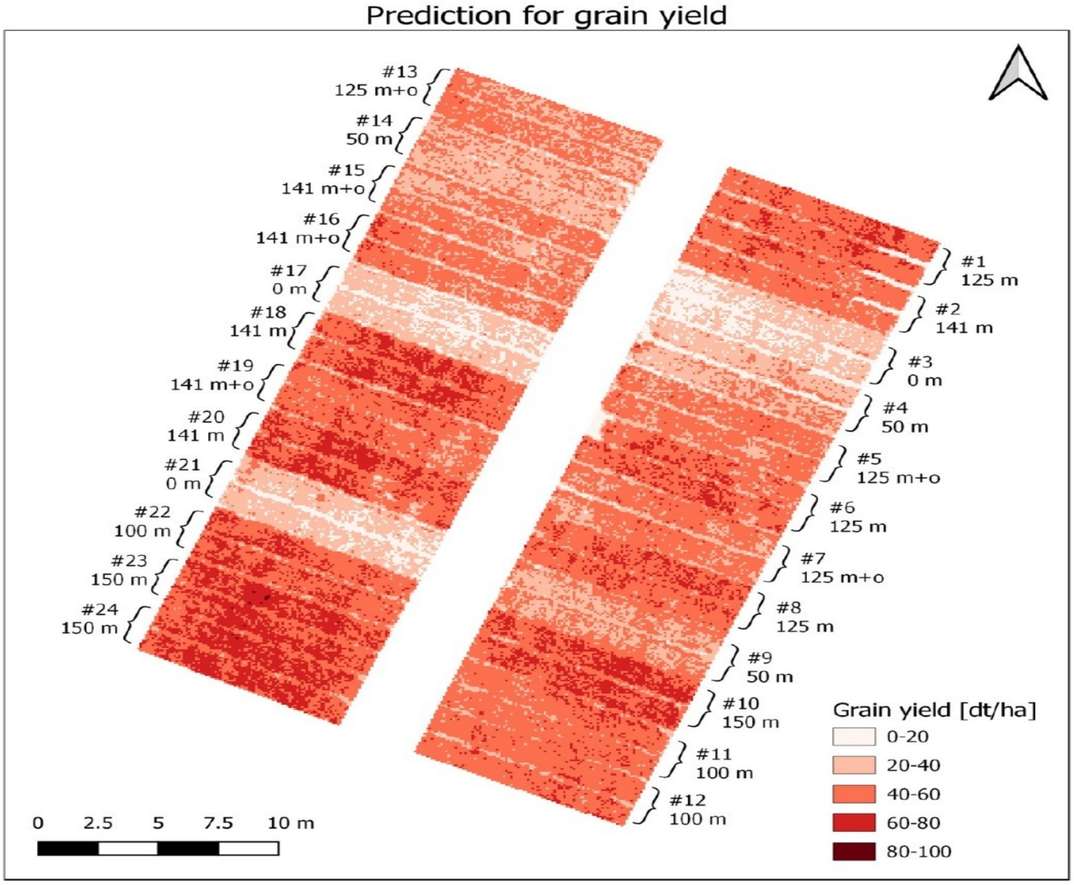


Figure 2.4: A false color image of predicted yield of wheat as a function of the fertilizer concentration [58].

The advancement in field deployable hyperspectral imaging systems has lead to interesting applications in environmental monitoring such as forest health tracking [84], water quality estimation [87], surface contamination, pollution and particulate monitoring [27, 94] and soil monitoring [91].

In medical imaging, hyperspectral images are being used as another modality, especially for disease diagnosis and image-guided surgery [72]. Medical hyperspectral imaging (MHSI) are employed for cancer detection, cardiology, pathology, retinal imaging, diabetic foot, shock, tissue pathology, mastectomy, gallbladder surgery, renal surgery, and abdominal surgery [43]. For example, figure 2.5 shows how three different types of white pills can be distinguished using hyperspectral images.

Hyperspectral machine vision can detect differences between materials more accurately, and more importantly, provide information not present in RGB wavelengths.

This is why they have found use in sorting, grading or process control in the machine industry.

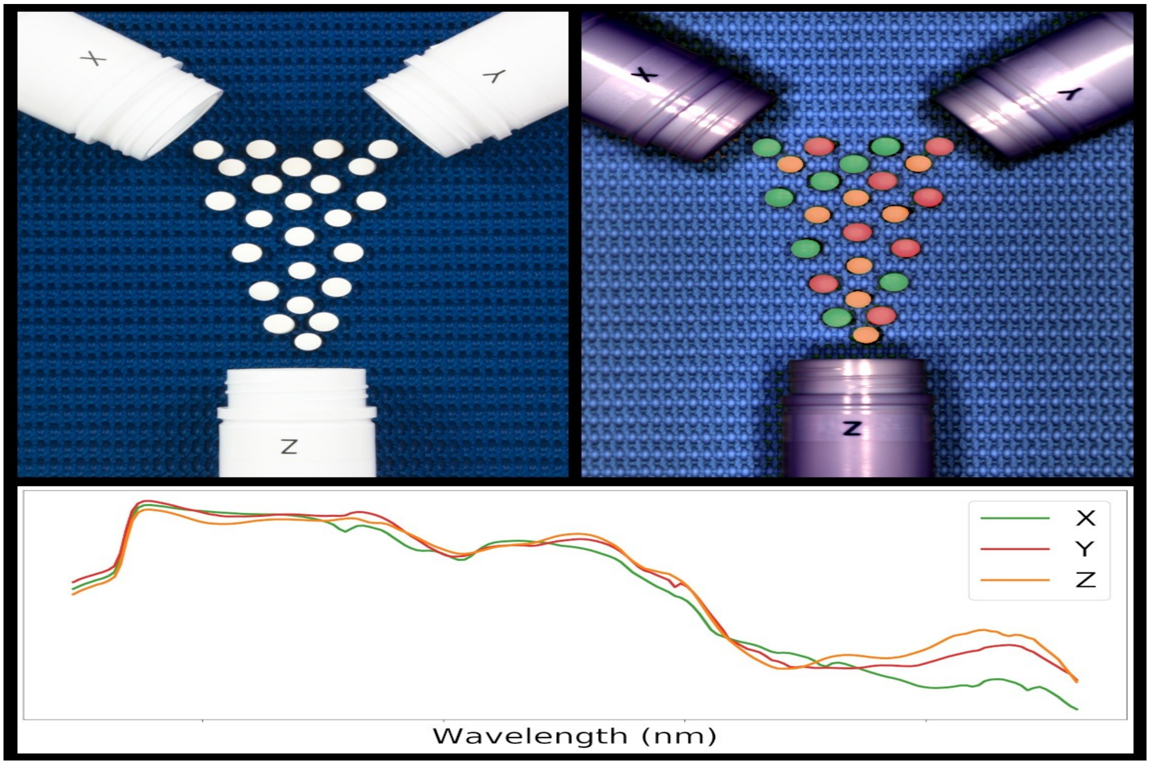


Figure 2.5: A false color image of three types of white pills, indistinguishable by color to the human eye, but accurately classified via near-infrared hyperspectral machine vision. [58]

The applications covered in this section first employ feature selection and extraction to reduce the data to a lower-dimensional feature space and find the most important features for further analysis. Then they employ supervised learning techniques to classify the data. Supervised learning techniques require the data to be labelled which needs extensive manual efforts. However, in this work we discuss unsupervised learning techniques which mitigates the above drawback.

## 2.2 Acquisition of Hyperspectral Datasets

Hyperspectral data is collected with an imaging spectroradiometer. Spectroradiometers measure the wavelength and amplitude of light. When directed towards a surface, they measure the light reflected in a range of wavelengths. Using an air-borne or satellite hyperspectral sensor, they measure spectral reflectance, which varies with the illumination and the physical properties of the surface being observed. Reflectance can be defined as the ratio of reflected energy to incident energy as a function of wavelength. However, spectral radiance is the variable actually measured by spectrometers. Spectral radiance depends on surface reflectance, spectrum of the input light, interactions of this energy during its downward and upward passages through the atmosphere, the geometry of illumination for individual areas on the ground, and characteristics of the sensor system [102]. These factors add noise and variability. The difference between radiance and reflectance is shown in Figure 2.6.

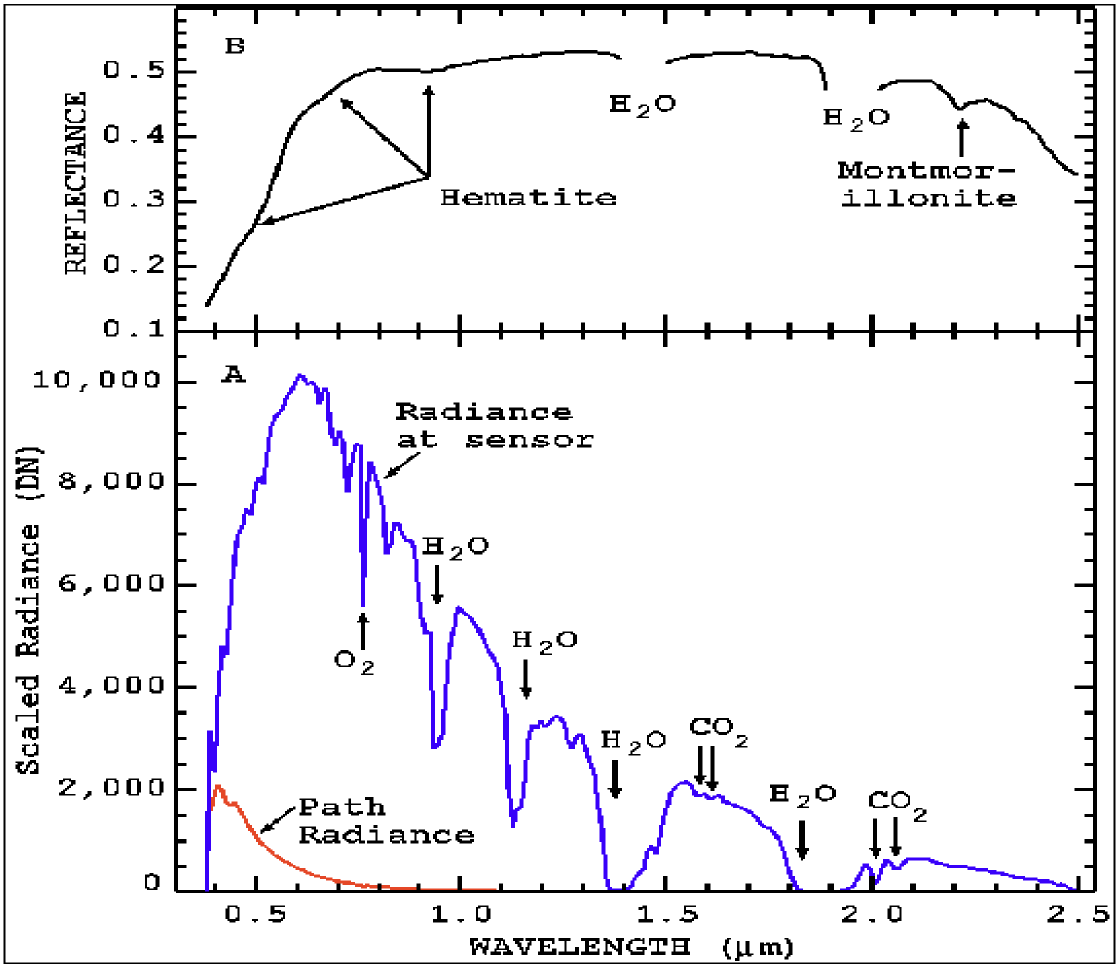


Figure 2.6: The figure depicts a radiance spectrum (bottom), of a mineral rich site, which is influenced by the solar function and absorption features caused by atmospheric gases. The radiance spectrum is calibrated and converted to surface reflectance (top). The reflectance spectrum contains the absorption feature(s) caused by minerals on the surface. Using the reflectance spectrum we can identify the materials, in this case hematite and montmorillonite. [4]

Therefore, there is a need to convert the obtained radiance to reflectance and accurate correction of atmospheric effects. There are many approaches to obtain reflectance such as scene-derived corrections, radiative transfer models, ground-calibration methods, and hybrid radiative-transfer-ground-calibration procedures [4].

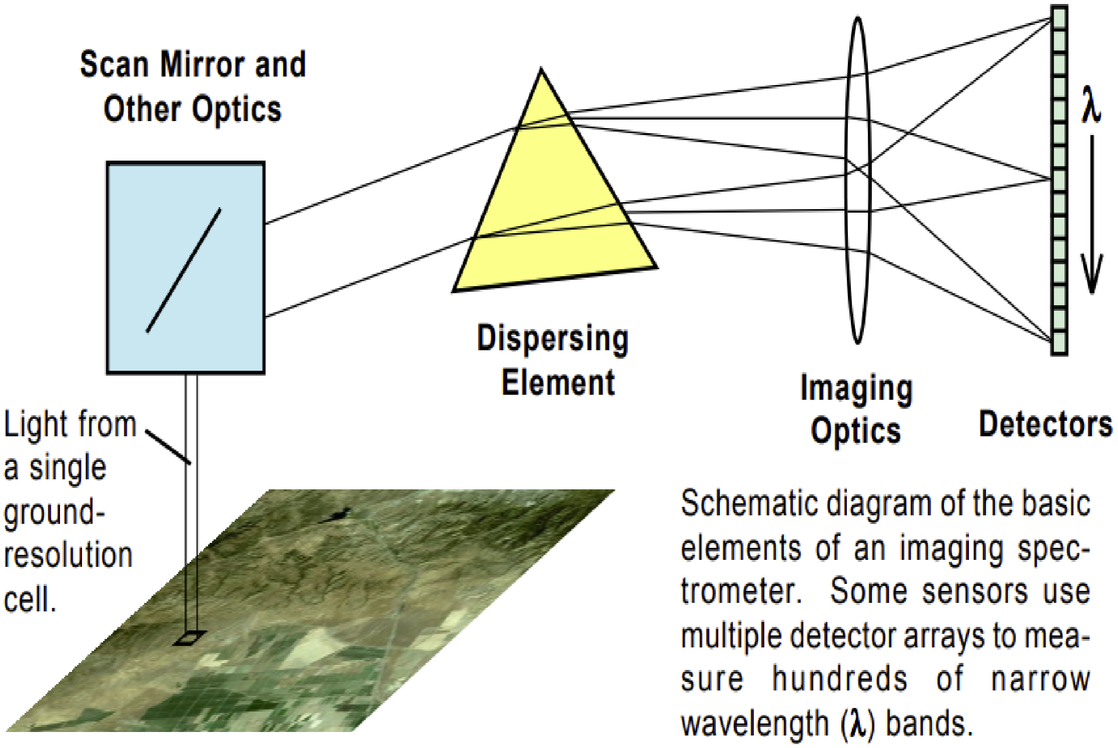

Scan Mirror and Other Optics
Dispersing Element
Imaging Optics
Detectors
λ
Light from a single ground-resolution cell.
Schematic diagram of the basic elements of an imaging spectrometer. Some sensors use multiple detector arrays to measure hundreds of narrow wavelength (λ) bands.

Figure 2.7: The schematic diagram of the basic elements of an imaging spectrometer are depicted. [102]

Within a spectrometer an optical dispersing element such a grating or prism is used to split the light. The splitting of light creates many narrow wavelength bands. The energy in each of these narrow bands is then collected by a detector. The measured radiance spectrum is multiplied by atmospheric effects to obtain the final reflectance.

## 2.3 Hyperspectral Datasets

In this thesis, we study and analyse several clustering techniques on hyperspectral datasets. The following sections introduce the hyperspectral datasets used in the experiments in the later sections.

### 2.3.1 Salinas-A

The Salinas-A dataset is 86 × 83 in size with 6 unique classes. It was collected by the 224-band AVIRIS sensor over Salinas Valley in California. The image has high spatial resolution of 3.7 meter pixels. This hyperspectral dataset contains information about vegetation, soil, and vineyards in the Salinas Valley.

### 2.3.2 Pavia Centre

Pavia Centre has 102 bands and is a 1096 × 1096 pixels image. It was acquired by the Reflective Optics System Imaging Spectrometer (ROSIS-3) sensor. We use a subset of the entire dataset i.e. only 6 unique classes, as in [80], for better comparative analysis. The classes present in the dataset are water, trees, asphalt, self-blocking bricks, bitumen, tiles, shadows, meadows, and bare soil.

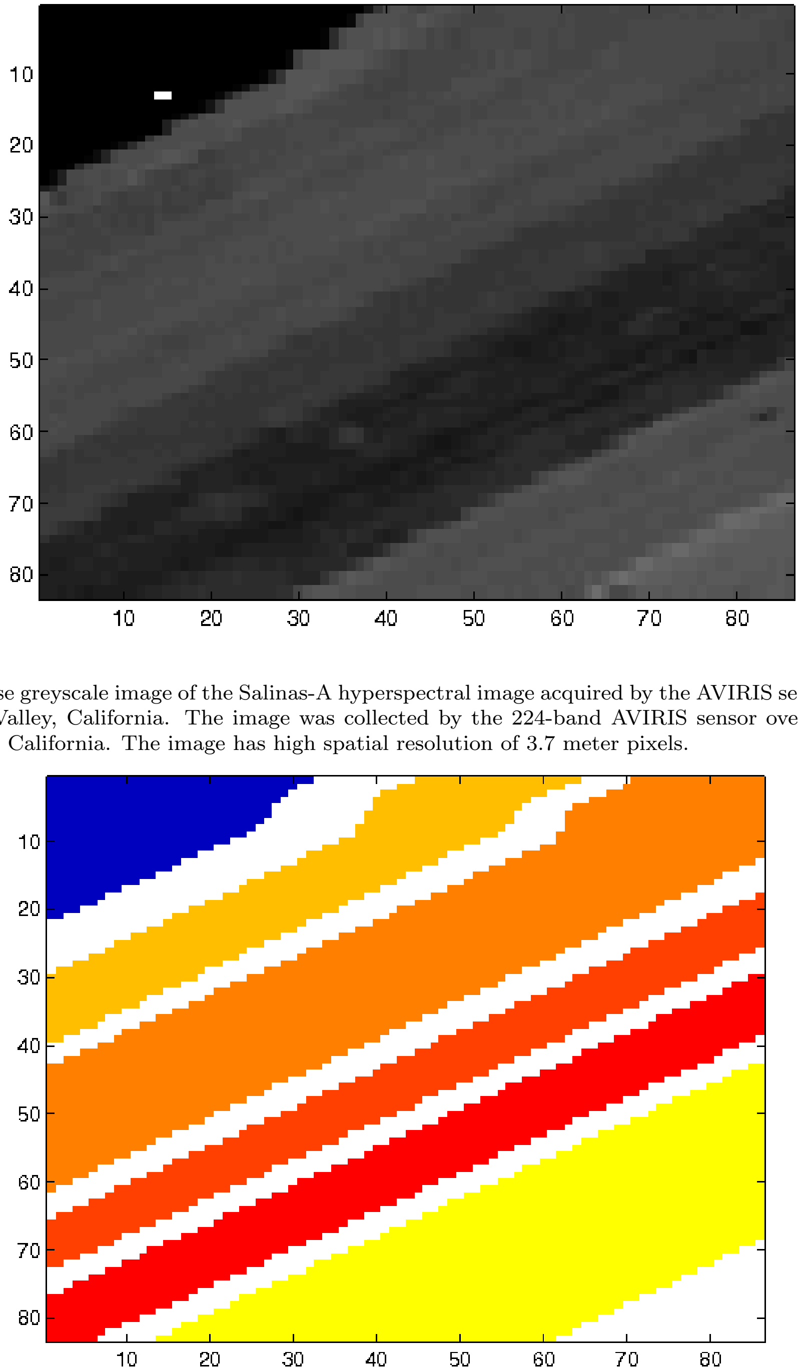
In Pavia Centre, there are samples that contain no information and have to be removed before any further analysis can be done.

### 2.3.3 Indian Pines

The Indian pines dataset is of size 145 × 145 with a spatial resolution of 20 m in a 2 mile by 2 mile area with 16 unique classes. It was acquired by AVIRIS on June 12, 1992 over Purdue University Agronomy farm northwest of West Lafayette and the surrounding area in Indiana, USA [12]. The data widely supported soils research at Purdue University. It contains 224 spectral bands ranging from 400 nm to 2.5 m. The hyperspectral classes are mainly of vegetation, forest and crops

The dataset was preprocessed to obtain clean spectral bands without noise and water absorption bands. Finally, 200 spectral bands were used in the experiments.

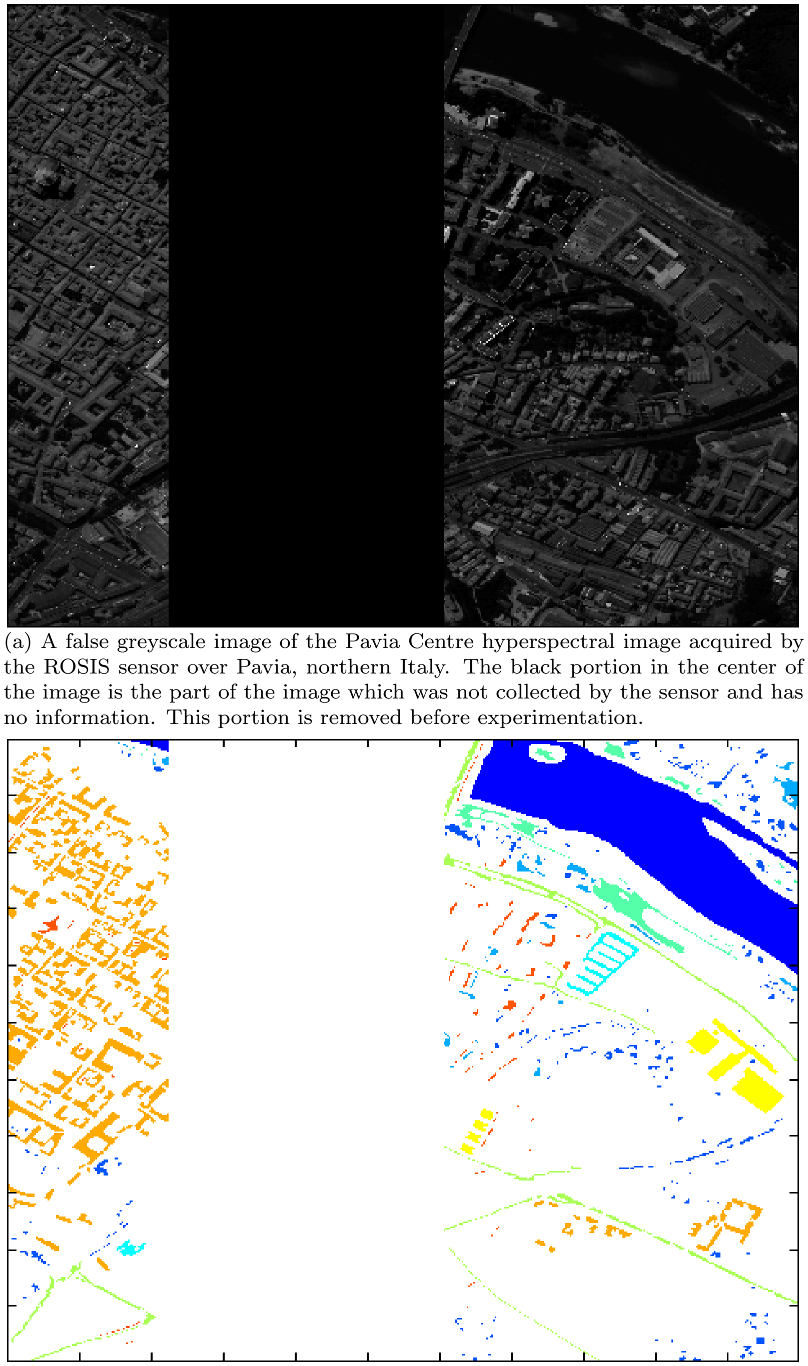
Figure 2.10 displays a false color image of the Indian Pines site. Figure 2.11 shows the ground truth image of the Indian Pines site which 16 distinct classes.



(a) A false greyscale image of the Salinas-A hyperspectral image acquired by the AVIRIS sensor over Salinas Valley, California. The image was collected by the 224-band AVIRIS sensor over Salinas Valley in California. The image has high spatial resolution of 3.7 meter pixels.

(b) Salinas-A ground truth map where the color scheme is indicative of the different classes present in the dataset. The classes represent vegetation, soil, and vineyards.

Figure 2.8: (a) shows the false greyscale image of the Salinas-A hyperspectral image and (b) shows the ground truth map.



(a) A false greyscale image of the Pavia Centre hyperspectral image acquired by the ROSIS sensor over Pavia, northern Italy. The black portion in the center of the image is the part of the image which was not collected by the sensor and has no information. This portion is removed before experimentation.

(b) Pavia Centre ground truth map where the color scheme is indicative of the different classes present in the dataset. The classes present in the dataset are water, trees, asphalt, self-blocking bricks, bitumen, tiles, shadows, meadows, and bare soil.

Figure 2.9: (a) shows the false greyscale image of the Pavia Centre hyperspectral image and (b) shows the ground truth map.

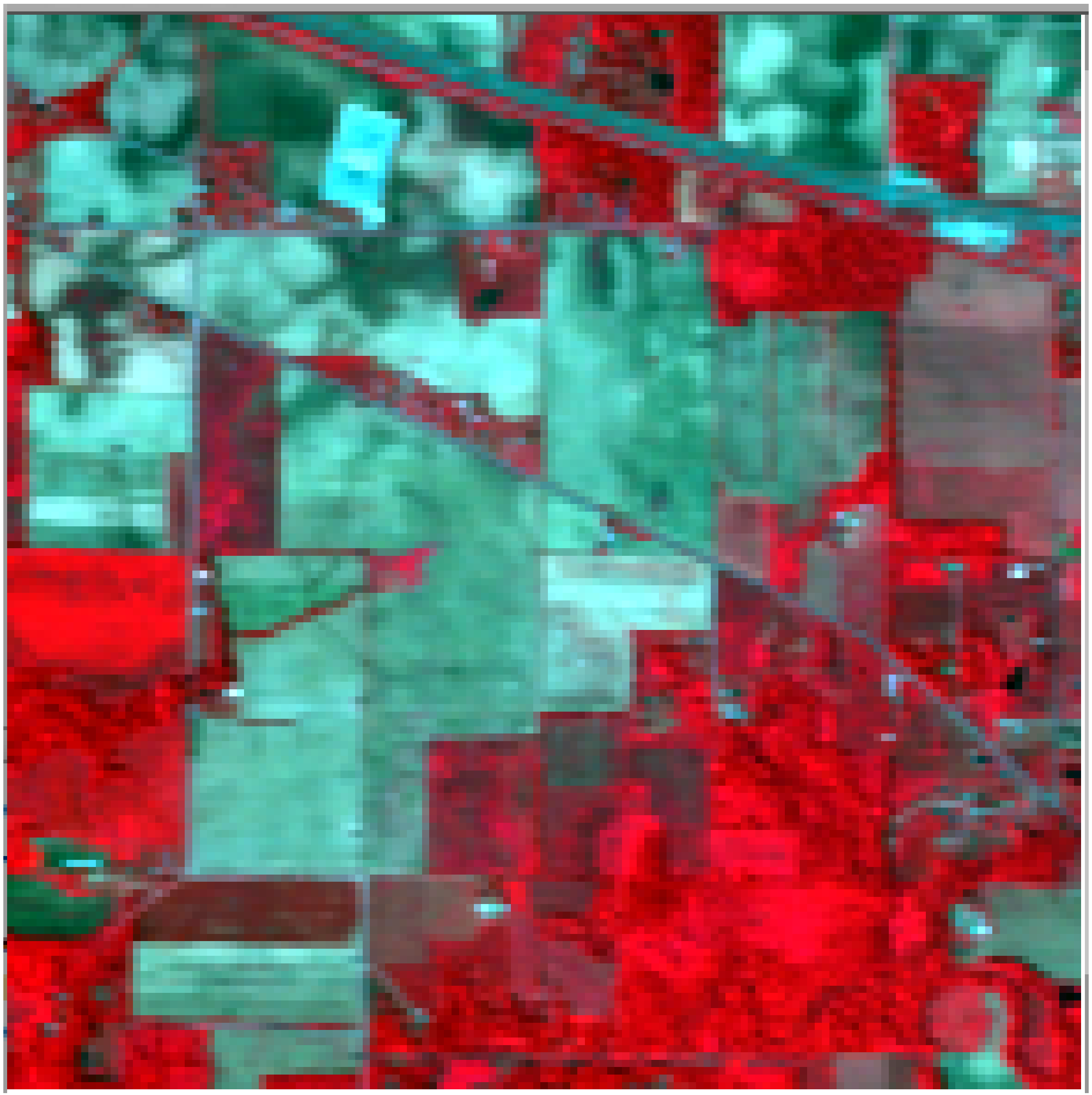


Figure 2.10: A false color image of the Indian Pine Site 3 hyperspectral image that was obtained by AVIRIS on June 12, 1992 over the Purdue University Agronomy farm. [12]

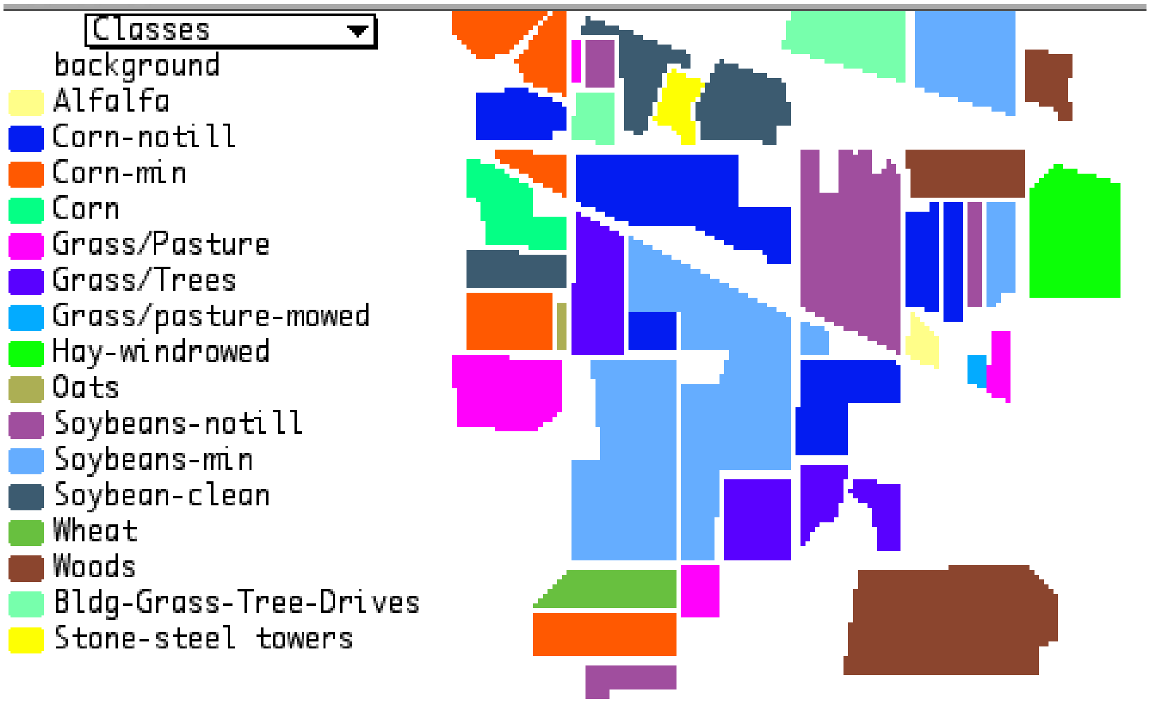


Figure 2.11: Indian Pine Site 3 ground truth map where the classes present in the hyperspectral image are given on the left side of the figure. [12]

## 2.4 Dataset Statistics

Table 2.1 gives the dataset statistics for the datasets used in this work. The datasets selected for experimentation are specifically chosen to show generality of the algorithms being studied and scrutinized. These datasets are considerably different in spatial and spectral resolutions, and also in terms of number of classes present.

Table 2.1: Dataset statistics for the hyperspectral datasets used in the experiments in this thesis.

| Dataset | Spatial Size | Spectral Size | Number of Classes |
|---|---|---|---|
| Salinas-A | 86 ×83 | 204 | 6 |
| Pavia Centre | 1096 ×1096 | 103 | 6 |
| Indian Pines | 145 ×145 | 200 | 16 |

The regions and scenes captured by the sensors to create these hyperspectral datasets, are also varied. We have scenes from urban areas, agricultural lands and a mixture of both. The spectra in these datasets are from different bands of wavelengths and have different absorption features. We would also like to note that the scenes captured in these images have varied geometric complexity. In the cropland dataset, Indian Pines, we observe clear highly structured geometries whereas this is not the same for the Pavia Centre dataset.

We also wish to empirically show the effects of change in spectral and spatial resolution, as well as the change in the number of ground truth classes. This is why the datasets used in this work are collected by a range of sensors i.e. AVIRIS, AVIRIS-NG and ROSIS which have varied spatial resolutions. The number of spectral bands present in the datasets also differ and so do the number of ground truth labels.

We wish to have a consistent comparative analysis with [80] and therefore use a subset of the Pavia Centre dataset in our experiments that has 6 ground truth classes.

There are several other hyperspectral datasets that could have been used. However, with other datasets, we face the problem of class overlap. For example, the mining dataset taken at the geologically well-studied site of Cuprite, Nevada, USA which was acquired by AVIRIS-Next Generation(NG) sensor and is the most diverse geologic dataset with more than 200 mineral classes. It has high class overlap due to the mixing of around 200 mineral classes whereas Pavia Centre dataset (considered in this work) only has 6 classes. The Cuprite dataset is better suited for tasks like feature extraction where the aim is to look for the most important features suited to represent the dataset.

# Chapter 3

# Related Work

In this chapter, we first discuss the related work for different clustering techniques, then for feature extraction techniques, especially dimensionality reduction techniques. Finally, we look at how clustering and dimensionality reduction techniques can be combined to produce a higher clustering performance.

## 3.1 Clustering Techniques

Clustering algorithms based on partitioning include $k$-means [59] and $k$-mediods [86]. In $k$-means clustering, the data points are clustered by defining centres of $k$ clusters and iteratively updating the centres and the labels for data points to meet a convergence criteria which achieves high inter-class variation and low intra-class variation. $K$-mediods is an improvement over $k$-means which efficiently deals with discrete data. These algorithms are still used for cluster analysis due to their low time complexity and high computational efficiency. However, the major drawbacks of these algorithms are that the data should be convex or the algorithm converges to a local optimal solution. Furthermore, the number of clusters $k$ needs to be known for each run of these algorithms.

In hierarchical clustering, the data is divided into clusters by establishing hierarchical relationships within the data. Data points are first clustered into individual clusters and then relationships between these clusters are defined to finally have one cluster that includes all individual clusters. Some of the commonly used hierarchical clustering algorithms include BIRCH [115], ROCK [50], CURE [51] and Chameleon [60]. BIRCH clusters data by building a feature tree where each node represents a subcluster. However, this tree will grow dynamically for each data point as it is added. An improvement over BIRCH is CURE that does not employ a feature tree. It uses random sampling to cluster a data point separately and adds that to the tree once it is clustered, and therefore is more suited for datasets with large number of sample points. On the other hand, ROCK further improves upon CURE and is able to handle enumeration type data. In the case of Chameleon, the input data points are first divided into smaller clusters using a nearest neighbor graph and then merged to make bigger clusters hierarchically. These algorithms are suitable for data that is present in non-convex and assume arbitrary shapes. They are also scalable but the time complexity suffers when higher number of clusters are present. Another disadvantage is the same as that with partitional clustering algorithms where the number of clusters needs to be known.

Clustering algorithms based on fuzzy theory use soft-assignment instead of hard class assignments and a data sample can belong to one or more clusters with a probability of $[0, 1]$. These algorithms help to cluster datasets where there is high class overlap and the cluster boundaries are weak and ambiguous. This set of algorithms is also capable of finding more complex and sophisticated relations between data points which may not be found in the ground truth labeling. FCM [36] is a fuzzy clustering algorithm. It was realized by Bezdek [14] where each data point is first assigned a membership to all possible clusters using an objective function. Through the years, several variants of FCM have been developed. FCS [30] and its variants are useful for the detection of curved boundaries, especially circular and elliptical. They use hyper-spherical-shells and hyper-ellipsoidal-shells as cluster prototypes, defined at the time of algorithm initialization. FCS based clustering algorithms cluster based on a distance measure corresponding to the shells. The Mountain Method (MM) [111] deals with the problem faced by FCM and FCS where it is difficult to find initial cluster centres. MM is an algorithm that uses a mountain function to approximately estimate the initial cluster centres. The major advantage of fuzzy theory based clustering algorithms is that they give a probability estimate for cluster assignment for each data point and therefore can achieve high clustering accuracies. However, these approaches do not scale for large datasets, often settle at a local minima, the clustering performance depends on the initial parameter assignments, and also the total number of clusters to be found by the algorithm needs to be preset. Ezzatabadi Pour and Homayouni use FCM and hyperspectral domain knowledge in terms of spectral similarity measures such as spectral angle (discussed in Section 4.3.3) to cluster datasets. Alhichri et al. propose an ensemble method using FCM and Markov Random Fields to cluster hyperspectral datasets. The experimentation based on fuzzy theory are in the scope of our future work.

The set of clustering algorithms based on distribution assume that the data points in a cluster belong to the same distribution. Gaussian Mixture Model (GMM) [5] is an example of this type of clustering algorithm that assumes that the original dataset is generated from a mixture of Gaussian distributions. GMM is sensitive to parameter initialization and has a high computational complexity. Furthermore, the assumption that data points are drawn from a mixture of Gaussian distribution is not always true and therefore GMMs are not applicable to all datasets.

Density based clustering algorithms are a set of clustering algorithms based on the idea that the clusters are areas of contiguous high density separated by contiguous low density regions. The low density regions are usually noise or outliers. The typical density based clustering algorithms are DBSCAN [38], OPTICS [11] and Mean-shift [25]. DBSCAN uses the basic idea of density based clustering. OPTICS improves upon DBSCAN and mitigates its disadvantages which are that DBSCAN is sensitive to two parameters i.e. the radius of the neighborhood and the minimum number of points in a neighborhood. In Mean-shift clustering, we first calculate the mean offset of the current data point and the next data point depends on the current data point which is then used to calculate the offset. Lastly, the iterations are continued until a convergence criteria is satisfied. Based on the kernel in the algorithm, the time complexity of Mean-shift clustering is high. The major advantage of density based clustering is that the data points can assume any arbitrary shape. On the other hand, density based clustering suffers from the following disadvantages; when the density space is uneven the algorithm has low clustering performance, memory efficiency is low for high data volumes and the algorithms are sensitive to parameters.

Algorithms like DL and DLSS [80] are density-based spectral-spatial techniques that combine geometric learning through a diffusion process [22]. The techniques employ diffusion distance [23] to exploit the non-linear and noisy relations present in the high-dimensional datasets by projecting them to a low-dimensional feature space. Diffusion geometry [22, 23] is used to identify the class modes which are then propagated to all the data points in the dataset through a nonlinear process that incorporates both spectral and spatial information. The spectral-spatial labeling technique is more robust than only employing spectral information to label the data. These techniques can also be thought of as clustering techniques that combine dimensionality reduction and clustering (discussed further in Section 3.3). We also discuss these techniques in detail in later chapters.

Clustering algorithms based on graph theory use a graph to represent the clustering problem where each node is a data point and the edge is the relationship between those data points. These methods do not make any prior assumptions about the clusters present in the original datasets i.e. number, size, density or the shape of the clusters. Meng et al. use graph-based clustering technique to cluster hyperspectral data. They do so without reducing the number of dimensions of the original data. A similarity graph based on pairwise comparisons of pixels is generated and is segmented using a pseudospectral algorithm that does not necessitate the creation of the full graph. Hufnagl and Lohninger employ a graph-based clustering technique which helps deal with class imbalance problem faced in many hyperspectral datasets and focuses on the analysis of minor features in the datasets. The authors also suggest that to cluster hyperspectral datasets, one must select two clashing methods to successfully cluster instead of two similar methods which can lead to interesting ensemble techniques in the future. However, as is mentioned in [55], graph-based techniques have a high time complexity as the graph complexity increases.

Spectral clustering is a sub-category of graph-based or graph partitioning clustering algorithms [16]. It represents a high-dimensional dataset using a low-dimensional space and clusters the data. A weighted graph is generated using a distance measure and then the Laplacian of this weighted graph is employed to cluster the data. Each vertex in the graph is assigned to a cluster based on similarities between data points in a cluster and dissimilarities between data points of different clusters. The main advantage of spectral clustering is that it does not make any strong assumptions about the statistics of the clusters unlike algorithms like $k$-means clustering. However, it is computationally inefficient for large datasets as the eigenvalues and eigenvectors of the similarity matrix need to computed which is a computational expensive process.

Manifold clustering is an unsupervised clustering technique where the data is represented as set of feature vectors in lower dimensional Euclidean space. These set of techniques assume that the data is intrinsically low-dimensional and can be represented by a lower dimensional space. The algorithms in turn perform non-linear dimensionality reduction. Local Linear Embedding (LLE) [92], Kernel Principal Component Analysis (KPCA) [46], and ISOMAP [110] are some of the manifold learning techniques. LLE embeds the data into a low dimensional space while still preserving the neighborhood information present in the data. However, LLE assumes that the manifold is convex. KPCA mitigates the issues with PCA by using kernels to find a non-linear embedding. ISOMAP improves on these algorithms by using geodesic distances and also preserving local neighborhood information. SMCE [37] is a manifold clustering technique that performs dimensionality reduction and clustering simultaneously. It finds a reconstruction matrix where data points are represented using an affine combination of $k$-nearest neighbors. It also adds a penalty on the distance on the reconstruction coefficient vector to ensure there are more zero values than non-zero values, also called a sparse penalty. This ensures that only the data points on the same manifold are given non-zero weights. However, SMCE leads to distortions in the global space which hinders the clustering performance. We empirically evaluate SMCE in later chapters.

## 3.2 Feature Extraction Techniques

It is to be noted that a good feature representation considerably aids in the clustering process [103]. Therefore, a task that is crucial for high clustering performance on high-dimensional hyperspectral datasets is the task of efficient feature extraction.

Hyperspectral images are high-dimensional data sets and experience the curse of dimensionality [34, 65]. Several algorithms fail to achieve high performance when the number of spectral dimensions is higher than the total number of data points [108]. Algorithms that employ a distance measure in 2D or 3D fail when applied to the high-dimensional hyperspectral datasets. Due to the lack of labelled hyperspectral datasets, there is a need for unsupervised methods for feature extraction and clustering of hyperspectral data sets. Dimensionality reduction techniques are used to extract important features which aid in clustering and also solve the problem of curse of dimensionality. These techniques should represent the high-dimensional data more efficiently in the lower-dimensional space and provide greater insights of the inherent data distribution. Algorithms like Principal Component Analysis (PCA) [89], Independent Component Analysis (ICA) [96], and Linear Discriminant Analysis (LDA) [19] are linear mapping dimensionality reduction techniques. However, these techniques are unable to capture the non-linearities present in hyperspectral datasets.

Deep learning techniques have come to be indispensable with the availability of large amounts of data and the use of GPUs. They have been applied by researchers to various fields and have several applications. However, these techniques require large datasets for training which is a limitation in remote sensing because transferring data back to Earth is costly. It is extremely difficult to create new ground truth datasets as it needs an understanding of the composition of the land being studied. This is the reason why there are few hyperspectral datasets that can be employed for research [81]. Current literature only uses a small set of datasets i.e. Pavia University, Salinas Valley and Indian Pines. Therefore, unsupervised dimensionality reduction techniques are highly applicable in this scenario.

Autoencoders [93] are used for deep learning based feature extraction from hyperspectral data [70, 76, 100]. Autoencoders are neural networks which are trained to reconstruct the output by using the input. Therefore, it is possible to encode the data using lesser dimensions which helps in feature learning. When non-linear functions are used for reconstruction, we can obtain a non-linear low-dimensional feature representation. There are several constraints that an autoencoder needs to adhere to. Due to this, an autoencoder can represent the input data in a condensed form. The condensed form is then used for feature representation. Several different reconstruction loss functions have been used in the literature, the most applied approach being sum of squared error (SSE). However, hyperspectral datasets contain spectral information. There are several alternative loss functions that employ the spectral information present in hyperspectral datasets. Windrim et al. use cosine of spectral angle as the reconstruction loss function for an autoencoder. Authors in Windrim et al. further employ spectral information divergence (SID) [18] and spectral angle [114] as the objective function in their autoencoder. We study these in detail in later chapters.

A hyperspectral image also contains spatial information. Spatial information gives several insights about the local features present around individual pixels and in turn improves clustering performance [42]. Hand-crafted spectral-spatial features such as extended morphological profile [41], morphological attribute profiles [29], and rotation invariant spectralspatial feature representation [99] can be used. However, these techniques are not suitable for all hyperspectral datasets and therefore are not very generalizable. Ensemble techniques using multiple kernels that exploit different features can solve this problem [15, 66, 67]. However, there are deep learning based data driven techniques that learn features hierarchically and therefore are more robust. Tao et al. propose a spectral-spatial feature learning framework based on multi-layer perceptron autoencoder for supervised classification of hyperspectral datasets which is more robust compared to hand-crafted features. Mou et al. suggest a convolutional autoencoder for feature learning to extract spatial features from hyperspectral images. This work looks at one-dimensional spectral information techniques. The use and assessment of spatial information techniques is part of the future work.

## 3.3 Combining Clustering and Feature Extraction Techniques

In the above Section 3.2, we looked at how dimensionality reduction techniques are used for feature extraction in the case of hyperspectral images. The next step in the pipeline is to cluster the data.

The first set of algorithms, cluster data using the clustering techniques discussed in Section 3.1. The second set of algorithms, combine the two steps i.e. dimensionality reduction and clustering. Each run of these algorithms will produce a clustering result and optimize it based on a definitive clustering criteria. Deep clustering techniques do so using an autoencoder framework [52, 109]. An autoencoder obtains a low-dimensional feature space and a clustering technique is applied to this feature space to obtain pseudo-labels for the data points. Then, the pseudo labels are used as supervision to update the encoder weights and the process continues until a convergence criteria is met. This strategy can easily corrupt the space topology as we continue to use hypothetical similarities from the pseudo labels. Therefore, the algorithm may compute discriminative features which do not match the discriminative features in the original dataset. This is referred to as Feature Randomness [77]. Autoencoders are therefore provided with their decoding and reconstruction capabilities to allow better reconstruction and less randomness. However, we face a trade-off between clustering and reconstruction. The reconstruction objective function preserves discriminative features whereas the clustering objective function gets rid of discriminative features. This problem is known as Feature Drift [77]. Mrabah et al. propose a deep learning technique to mitigate the above two issues by employing a dynamic loss function that gradually and smoothly changes a self-supervised objective function to a pseudosupervised objective function which achieves greater clustering performance. We would like to apply this technique to hyperspectral datasets as part of our future work.

## 3.4 Hyperspectral Super-Resolution Techniques

In the above sections, we look at clustering and feature extraction techniques. These techniques apply to data with high spatial resolution but do not perform as well when the spatial resolution is low as is the case for hyperspectral data. Therefore, we introduce a new concept called hyperspectral super-resolution.

Hyperspectral images have high spectral resolution but low spatial resolution due to hardware level fundamental physical limitations [63]. In contrast, conventional RGB images have higher spatial resolution and lower spectral resolution as they integrate the radiance across a wide wavelength range. The low spatial resolution of hyperspectral images limits its use [20, 61] in many fields. Therefore, we can benefit by understanding the underlying joint spatial-spectral structure present of a hyperspectral image.

Hyperspectral super-resolution is used to fuse a hyperspectral image that has a high spectral resolution, with a conventional image that has a high spatial resolution to obtain an image with high spectral and spatial resolution. It can be considered as being related to multi-spectral pan-sharpening where a low resolution multi-spectral image is fused with a high resolution panchromatic image [71]. Matrix factorization is utilized to fuse RGB or multi-spectral images with hyperspectral images [7, 8]. Bayesian representation is also used for hyperspectral super-resolution [6, 8]. The third category of super-resolution techniques for hyperspectral datasets employs tensor factorization [32, 68]. These techniques use hand-crafted priors like low-rankness and sparsity while formulating an optimization problem for the super-resolution fusion process [45]. Instead of using hand-crafted priors, we can use convolutional neural network (CNN) based techniques for hyperspectral super-resolution [33, 88]. Fu et al. propose an unsupervised CNN based hyperspectral super-resolution technique to understand the underlying characteristics of hyperspectral images. The authors also study the effect of RGB camera spectral response (CSR) functions for HSI super-resolution which improves hyperspectral super-resolution performance.

# Chapter 4

# Survey of Clustering Methods

In this chapter, we survey machine learning techniques to classify hyperspectral data in an unsupervised fashion.

## 4.1 $k$-means Clustering

In unsupervised learning, inferences about the data are made based solely on the data and not any prior information or external guidance. The algorithm is not provided any labels in the input data.

The objective of $k$-means clustering is to group data points into $k$ clusters where each data point is assigned a cluster whose centroid is closest to the data point [59]. Given $n$ data points in a $d$-dimensional space, the algorithm groups these $n$ data points into $k$ clusters where $k$ is less than or equal to $n$. The algorithm aims at minimizing the squared error function:

$$\arg\min \sum_{i=1}^{k} \sum_{x \in c_i} ||x - \mu_i||^2 \tag{4.1}$$

where $\mu_i$ is the centroid of the cluster $c_i$ and $x_1, x_2, ..., x_n \in c_i$ are the $n$ data points.

The steps for the $k$-means clustering algorithm are as follows:

1. Randomly select $k$ cluster centroids.

2. Calculate the distance between every data point and the $k$ cluster centroids.
3. Assign a data point to a cluster based on the shortest distance to the centroid of that cluster
4. Calculate the mean of the data points in a cluster which will be the new cluster centroids
5. Repeat steps 2, 3 and 4 until no data points are reassigned.

One of the major disadvantages of $k$-means clustering is that it finds the local minima of the objective function defined in equation 4.1. Furthermore, the solution to the algorithm depends on the initialization of the cluster centroids. $k$-means clustering also fails for datasets that do not have spherical clusters.

## 4.2 Principal Component Analysis

Principal component analysis (PCA) is a data analysis technique that transforms a large number of correlated variables into a small number of uncorrelated variables called the principal components [98]. It is a dimensionality reduction method which can help to understand the underlying structure in a complex data set. PCA does so by finding variables that are linearly independent of each other, a linear transform of the data. Every principal component is orthogonal to every other principal component.

Let $X$ be an $n \times d$ matrix where $n$ represents the number of data points, each with $d$ number of real valued features. The steps involved in PCA are as follows:

1. Normalize the data set so that the mean of the data points is now zero and the standard deviation is one, forming the matrix $Z$.
2. Find the covariance matrix of $Z$ which is $\Sigma = Z^T Z$.
3. Find the eigenvectors of $\Sigma$ using eigen decomposition where $\Sigma$ can be factorized as $PDP^{-1}$. The columns of $P$ matrix are the eigenvectors of $\Sigma$ and the diagonal elements of the diagonal matrix $D$ are the corresponding eigenvalue.

   Or we can find the eigenvectors of $\Sigma$ using singular value decomposition (SVD) i.e. $[U, S, V] = svd(\Sigma)$, which is computationally more efficient. The eigenvectors are the columns in the matrix $U$.
4. Finally, rearrange the eigenvalues in order of largest to smallest and consequently

also rearrange the corresponding eigenvectors. Now, pick the top $k$ eigenvectors where $k \leq d$ which will be the principal components.

## 4.3 Autoencoders

Autoencoders have been proposed as an unsupervised method for pre-training artificial neural networks (ANNs) [95]. They aid in representation learning by compressing the original data by finding patterns in the input features which are related or dependent. The aim of an autoencoder is to learn a function to output $\hat{x}$ when given the input $x = \{x_1, x_2, ..., x_m\}$, where each $x_i \in \mathbb{R}^n$ and each $\hat{x}_i \in \mathbb{R}^n$ , under certain constraints. More specifically, a neural network is designed where the input and output layer, both have $m$ elements, whereas the hidden layer(s) have $k$ ($k < m$) nodes.

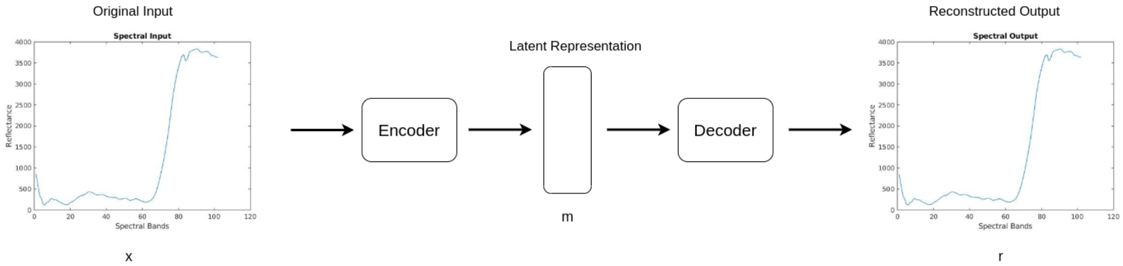


Figure 4.1: Architecture of an Autoencoder

If linear activation functions are used to build an autoencoder, then it performs dimensionality reduction similar to the PCA discussed in Section 4.2. However, if non-linear activation functions are used then an autoencodoer is capable of discovering more interesting features in the data.

A well designed autoencoder should accurately reconstruct the data. As the input is changed, the autoencoder should be sensitive enough to change the reconstruction accordingly. However, the autoencoder should not overfit to a single input dataset. The loss function used in the construction of an autoencoder aims to be sensitive to the input and insensitive to overfitting, as is given below,

$$\mathbb{L}(x, \hat{x}) + regularizer \tag{4.2}$$

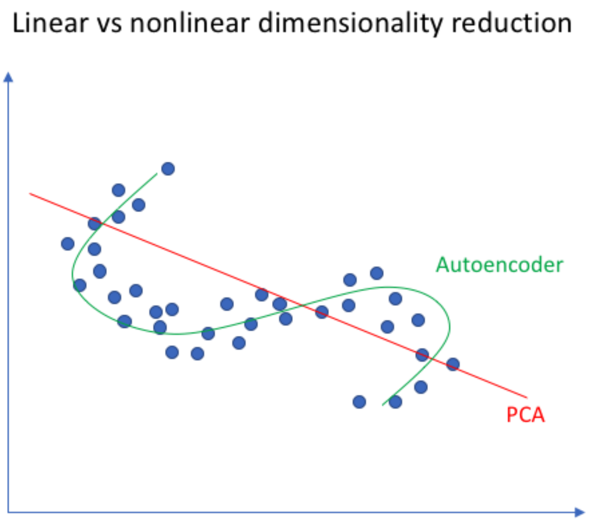


Figure 4.2: Difference between PCA and Autoencoder [1] is that PCA performs linear dimensionality reduction whereas an autoencoder is a nonlinear dimesionality reduction technique. In this figure, we see that the autoencoder is able to recognize the nonlinearities in the data whereas PCA is not.

### 4.3.1 Sparse Autoencoders

A sparsity constraint ensures that the final output contains more zeros than non-zero values. A sparse autoencoder is an autoencoder that uses the sparsity penalty. In a sparse autoencoder, the constraint on the neural network is not to have hidden layers with fewer number of nodes. A sparse autoencoder instead uses a loss function which penalizes the activations for a particular layer. In most cases, the weights are regularized but here we regularize the activations. So, the encoder and the decoder learn with fewer number of neurons in the hidden layers. Such an autoencoder is able to minimize memorization of the input data and extract more features for the latent space representation.

The sparsity constraint can be implemented in the following two ways:

1. L1 regularization: The regularization term penalizes the absolute value of the vector of activations $a$ in layer $h$ for observation $i$ as given below:

$$\mathbb{L}(x, \hat{x}) + \lambda \sum_i |a_i^{(h)}| \tag{4.3}$$

where $\lambda$ is the regularization parameter.

2. KullbackLeibler (KL) divergence: KL divergence or relative entropy is the measure of how one probability distribution is different from another probability distribution. If we have two discrete probability distributions $P$ and $Q$ that are defined on the same probability space, then the KL divergence between them can be written as:

$$D_{KL}(P||Q) = -\sum_{x \in \mathbb{X}} \log \frac{Q(x)}{P(x)} \tag{4.4}$$

Now, looking at how KL divergence can be used as a sparsity constraint for an autoencoder, let us have a sparsity parameter $\rho$ that denotes the average activation of a neuron for a number of data points and can be mathematically written as:

$$\hat{p_j} = \frac{1}{m} \sum_i [a_i^{(h)}(x))] \tag{4.5}$$

where $j$ is a particular neuron in the hidden layer $h$ and $m$ is the total number of training samples, denoted individually as x.

The loss function using KL divergence is as follows:

$$\mathbb{L}(x, \hat{x}) + \sum_i D_{KL}(\rho||\hat{p_j}) \tag{4.6}$$

In equation 4.6, $\rho$ is the reference probability distribution. In this case, $\rho$ is a Bernoulli random variable distribution and is used to compare the observed distribution $\hat{p_j}$. Conceptually, when the average activation of a neuron over a number of data points are constrained, we force the neuron to launch for only a subset of the data points.

### 4.3.2 Denoising Autoencoders

In a denoising autoencoder, noise is added to the input to the autoencoder but reconstruction loss is computed against the original data. Thus, the network learns a low-dimensional manifold which accurately represents the original data, without the added noise.

### 4.3.3 Autoencoders using Remote-Sensing Measures

In this section, we discuss autoencoders that can use remote-sensing measures in their loss functions. These loss functions are a way to incorporate domain knowledge which in turn changes the latent space obtained. We will look at two autoencoders the Spectral Information Divergence (SID) autoencoder that uses SID as its loss function and Spectral Angle (SA) autoencoder that employs SA as its loss function [108].

#### Spectral Angle Autoencoder

Spectral Angle (SA) is used to compute the similarity between two vectors. This angle helps to measure the difference in the shape of spectra instead of the magnitude and incorporates essential spectral information present in a spectrum. This measure is invariant of the brightness or the intensity of the spectrum. We employ this measure in the loss function for an autoencoder instead of sum of squared error (SSE). Therefore, the autoencoder captures the shape of the spectrum instead of the intensity of the spectrum.

The spectral angle $\theta_{SA}$ for two vectors $A$ and $B$ of size $d$ dimensions can be calculated as follows:

$$\theta_{SA} = \cos^{-1} \frac{\sum_{i=1}^{d} A_i B_i}{|A||B|} \tag{4.7}$$

We also introduce another autoencoder that employs the cosine of spectral angle (CSA) and it follows the following objective function:

$$\cos(\theta_{SA}) = \frac{\sum_{i=1}^{d} A_i B_i}{|A||B|} \tag{4.8}$$

#### Spectral Information Divergence Autoencoder

Spectral Information Divergence (SID) is a measure of the probabilistic variation between two spectra. It is an information-theoretic measure which we apply to the loss function of an autoencoder. SID proves to be better than SA as it is efficient in capturing the spectral properties and employ spectral variability into the autoencoder more effectively.

Let us look at two one-dimensional spectra, $A$ and $B$ of dimension $d$ channels or bands. Then, the SID can be computed as follows:

$$\mathrm{SID}(A,B) = \sum_{i=1}^{d} p_i \log \frac{p_i}{q_i} + \sum_{i=1}^{d} q_i \log \frac{q_i}{p_i} \tag{4.9}$$

where $p$ and $q$ are vectors of the normalized spectra $A$ and $B$:

$$p = \frac{A}{\sum_{t=1}^{T} A_t} \tag{4.10}$$

$$q = \frac{B}{\sum_{t=1}^{T} B_t} \tag{4.11}$$

where $A_t$ and $B_t$ are the elements of $A$ and $B$ for the corresponding spectral values at channel $t$ and $T$ is the total number of elements in $A$ or $B$.

Windrim et al. explain how SA and SID are incorporated into the loss function of the autoencoder and provide the derivatives required for back propagation and the parameter updates for gradient descent optimization [108].

## 4.4 Combining dimensionality reduction and k-means clustering

Often, a data set is reduced to a lower dimension to aid with the clustering process. PCA is a linear dimensionality reduction technique and an autoencoder is a non-linear one (by the assumption that non-linear activation functions were employed). Therefore, in a typical pipeline, first dimensionality reduction takes place which outputs a feature space. This feature space is the input given to a clustering algorithm. In the experiments conducted in this work, we combine PCA and autoencoders with k-means clustering. Using the new feature space which has a lower dimension than the original data set, also helps in visualizing the final clustering output better.

De la Torre and Kanade introduce Discriminant Component Analysis (DCA) which is an improvement over PCA with k-means clustering as it uses discriminative features for clustering instead of generative ones [31]. Ye et al. analyse the DisCluster framework that successfully integrates subspace selection and clustering. Both of these project the original data into a low dimensional space and then maximize the inter-cluster variance in order to find clusters.

## 4.5 Spectral Clustering and Sparse Manifold Clustering and Embedding

Sparse Manifold Clustering and Embedding (SMCE) [37] is a clustering algorithm that can simultaneously perform dimensionality reduction and clustering for data that lies in multiple nonlinear manifolds. Firstly, the algorithm finds a small neighborhood around each data point and appropriate weights are used to connect it to its neighbors. SMCE does so by solving a sparse optimization problem that finds the neighbors and the weights automatically. Finally, the solution of the optimization problem is used for dimensionality reduction and clustering; and spectral clustering and embedding is employed to do so.

Let there be $N$ data points $\{x_i \in \mathbb{R}^D\}_{i=1}^N$ that lie on $n$ different manifolds $\{\mathcal{M}_l\}_{i=1}^n$ of intrinsic dimension $\{d_l\}_{i=1}^n$. The algorithm assumes that each data point has a small number of neighbors that span a low-dimensional affine subspace. The neighborhood thus is given by the points from the same manifold. Now, the aim of the optimization algorithm is to find a few neighbors for each data point $x_i$ that lie in the same manifold $\mathcal{M}_l$. The neighborhood $\mathcal{N}_i$ of a point $x_i$ is considered to be of arbitrary size. The sparse optimization program is biased to find data points that are close to $x_i$ and span a low-dimensional affine subspace passing near $x_i$. Let there be some points $\{x_j\}_j$. The points that are neighbors of $x_i$ can be given by solving the following:

$$\|[x_1 - x_i \dots x_N - x_i]c_i\|_2 \leq \epsilon \text{ and } \mathbf{1}^T c_i = 1 \tag{4.12}$$

where for all $i$ there exists $\epsilon \geq 0$. The solution $c_i$ has $d_l + 1$ non-zero values which corresponds to the $d_l + 1$ neighbors of $x_i$ in $\mathcal{M}_l$.

From solving the above equation, we obtain the neighbors of data point $x_i$ and the weight vector $\boldsymbol{w}_i^T \triangleq [w_{i1} \dots w_{iN}] \in \mathbb{R}^N$ associated with $x_i$. The weights $\boldsymbol{w}_i$ are then used for dimensionality reduction and clustering. A similarity graph $\mathcal{G} = (V, E)$ is constructed with nodes representing the data points. The edge of the graph is given by $|w_{ij}|$, where node $i$ representing point $x_i$ connects to node $j$ representing point $x_j$. The similarity matrix of graph $\mathcal{G}$ can be written as:

$$\boldsymbol{W} \triangleq [|\boldsymbol{w}_1| \cdots |\boldsymbol{w}_N|] = \begin{bmatrix} \boldsymbol{W}[1] & 0 & \cdots & 0 \\ \boldsymbol{0} & \boldsymbol{W}[2] & \cdots & \boldsymbol{0} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{W}[n] \end{bmatrix} \boldsymbol{\Gamma} \tag{4.13}$$

where $\boldsymbol{W}[l]$ is the similarity matrix of the data points in $\mathcal{M}_l$ and $\boldsymbol{\Gamma} \in \mathbb{R}^{N \times N}$ is an unknown permutation matrix. Spectral clustering [83] is used to cluster the data by employing the $\boldsymbol{W}$ similarity matrix. The adjacency matrix $\boldsymbol{W}[i]$ of the $i$-th cluster can be used as a similarity between points in the corresponding manifold to obtain a low-dimensional embedding. Further details and reasoning behind the algorithm can be found in [37].

## 4.6 Deep Embedded Clustering

Deep Embedded Clustering (DEC) [109] focuses on obtaining a feature representation $Z$ of a data set $X$ and simultaneously clusters the data. Therefore, in a way the feature representation is forced to cater to the clustering loss. A deep neural network (DNN) is used to learn the optimum mapping and trained using Stochastic Gradient Descent (SGD). This algorithm also scales well for larger datasets as it has linear dependence on the number of data points.

The two phases in the DEC algorithm are:

1. Parameter initialization with a deep autoencoder
2. Parameter optimization or clustering in the process iterates between defining an auxiliary target distribution and minimizing the KL divergence to it.

The initial estimate of the non-linear mapping $f_\theta$ and the initial cluster centroids $\{\mu_j\}_{j=1}^{k}$ are given.

Now, the second phase involves the following two steps which are repeated till a convergence criteria is met:

### Soft Assignment

In this step, a soft assignment between the embedded points and the cluster centroids is computed. The Students t-distribution is used as a kernel to measure the similarity between embedded point $z_i$ and centroid $\mu_j$ as given in equation 4.14.

$$q_{ij} = \frac{\left(1 + \|z_i - \mu_j\|^2 / \alpha\right)^{-\frac{\alpha+1}{2}}}{\sum_{j'} \left(1 + \|z_i - \mu_{j'}\|^2 / \alpha\right)^{-\frac{\alpha+1}{2}}} \tag{4.14}$$

where $z_i = f_\theta(x_i) \in Z$, $x_i \in X$, $\alpha$ are the degrees of freedom of the Students t-distribution and $q_{ij}$ can be interpreted as the probability of assigning sample $i$ to cluster $j$ (therefore it is a soft assignment).

### KL Divergence Minimization

The loss function for updating the deep mapping $f_\theta$ is the KL divergence between the soft assignment $q_i$ and the auxiliary target distribution $p_i$ as:

$$L = \mathrm{KL}(P\|Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}} \tag{4.15}$$

It is necessary to pick the appropriate target distribution $p_i$. It should have the following properties:

- Improve cluster purity.
- Data points assigned with a higher confidence should be taken more into account.
- The loss contribution of each centroid is normalised so that the large clusters do not distort the feature space

The target distribution $p_i$ is computed with the following formula:

$$p_{ij} = \frac{q_{ij}^2 / f_j}{\sum_{j'} q_{ij'}^2 / f_{j'}} \tag{4.16}$$

where $f_j = \sum_i q_{ij}$ are soft cluster frequencies.

The cluster centers $\{\mu_j\}$ and DNN parameters $\theta$ are jointly optimized using Stochastic Gradient Descent (SDG) with moment.

$$
\begin{aligned}
\frac{\partial L}{\partial z_i} =& \frac{\alpha+1}{\alpha} \sum_j \left(1 + \frac{\left\|z_i - \mu_j\right\|^2}{\alpha}\right)^{-1} \\
& \times \left(p_{ij} - q_{ij}\right)\left(z_i - \mu_j\right) \\
\frac{\partial L}{\partial \mu_j} =& -\frac{\alpha+1}{\alpha} \sum_i \left(1 + \frac{\left\|z_i - \mu_j\right\|^2}{\alpha}\right)^{-1} \\
& \times \left(p_{ij} - q_{ij}\right)\left(z_i - \mu_j\right)
\end{aligned}
\tag{4.17}
$$

Standard backpropogation is used to trickle down the gradients $\frac{\partial L}{\partial z_i}$ to compute the DNN's parameter gradient $\frac{\partial L}{\partial \mu_j}$. When less than $tol\%$ of data points change their cluster assignments, the procedure is stopped.

The first phase in DEC is parameter initialization. Firstly, DEC is initialized with a Stacked Autoencoder (SAE) [105]. Each layer in the SAE is a denoising autoencoder trained on the previous layer's output after adding random noise.

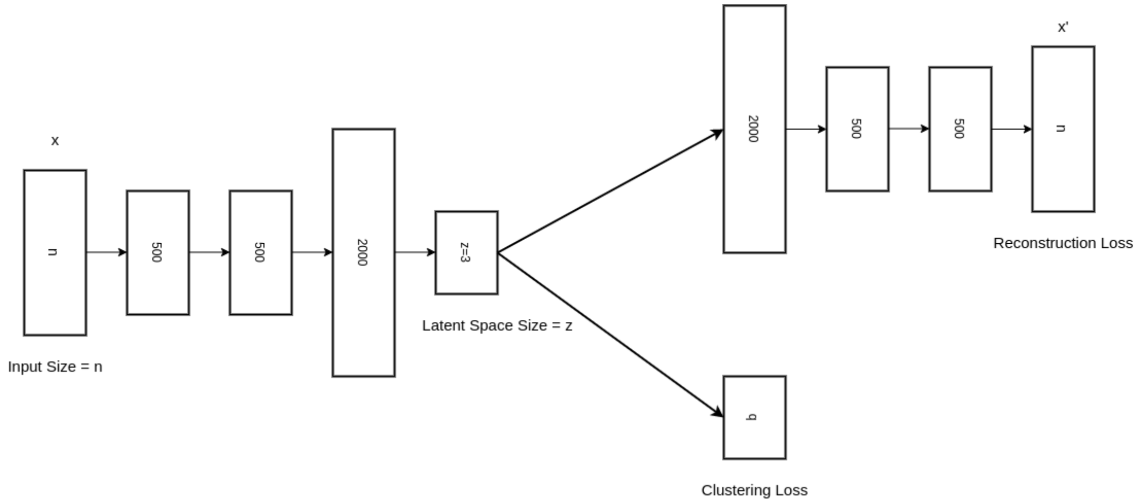


Figure 4.3: Architecture of Deep Embedded Clustering (DEC). We see that the encoder and decoder have the same number of layers and the number of neurons in these layers are 500, 500, 2000.

A denoising autoencoder has two layers in the neural network which can be defined as:

$$
\begin{aligned}
\tilde{x} \sim& \operatorname{Dropout}(x) \\
h =& g_1\left(W_1 \tilde{x} + b_1\right) \\
\tilde{h} \sim& \operatorname{Dropout}(h) \\
y =& g_2\left(W_2 \tilde{h} + b_2\right)
\end{aligned}
\tag{4.18}
$$

where *Dropout*() randomly sets a part of the input dimensions to 0, $g_1$ and $g_2$ are activation functions for encoding and decoding layer respectively, and $\theta = W_1, b_1, W_2, b_2$ are model parameters. Least squares loss i.e. $||x - y||_2^2$ is employed for training the network. Rectified linear units (ReLUs) are used in all of the encoder and decoder pairs. $g_2$ of the last layer does not use ReLU as for reconstruction of the input data we require both the negative and the positive values. Also, $g_1$ of the layer which yields the final feature representation i.e. the layer in the middle does not use ReLU so that the final feature representation or data embedding has full information.

The initial cluster centroids $\{\mu_j\}_{j=1}^k$ are obtained by obtaining the feature representation $Z$ and performing k-means clustering in the $Z$ space.

## 4.7 Gaussian Mixture Model

Gaussian Mixture Model (GMM) [5] is a clustering technique that mitigates the problem of hard assignments. Hard assignment means that every data point is assigned to one and only one cluster. However, GMM uses soft assignments or in other words gives a probability measure that a data point belongs to a particular cluster.

A Gaussian mixture is a function that constitutes a mixture of $k \in \{1, \ldots K\}$ Gaussians, where $k$ is the number of clusters present. The $k^{th}$ Gaussian is parameterized with the following parameters:

- $\mu$: A mean that defines cluster center
- $\Sigma$: A covariance that defines the width of a cluster. In a multivariate Gaussian, the covariance represents the dimensions of an ellipsoid.
- $\pi$: The mixing probability that defines how big or small the Gaussian function will be. Also, $\pi$ must meet the condition $\sum_{k=1}^{K} \pi_k$.

To find the optimal parameter, we employ maximum likelihood of the Gaussian density function.

$$\mathcal{N}(\mathbf{x}|\mu,\Sigma) = \frac{1}{(2\pi)^{D/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x}-\mu)^T\Sigma^{-1}(\mathbf{x}-\mu)\right) \tag{4.19}$$

where, $x$ represents the data points, $N$ is the total number of data points, $D$ is the dimension of a data point $x$.

We then take the derivative of equation 4.19 and equate it to zero, which gives the Maximum Likelihood Estimate (MLE), and will find the optimal values of the parameters $\mu$ and $\Sigma$.

## 4.8 Non-Negative Matrix Factorization

Non-negative Matrix Factorization (NMF) has proven useful in the fields of imaging, text mining and hyperspectral imagery [48] due to its ability of successfully extract sparse and relevant features from non-negative data vectors.

Let $X \in \mathbb{R}^{p\times n}$ be the data given, where $n$ are the total number of pixels of dimension $p$. NMF approximates the matrix $X$ to a low-rank approximation $X \approx WH$. The $p$ dimensions are reduced to $r$ i.e. $p > r$ to produce $W \in \mathbb{R}^{p\times r}$ and $H \in \mathbb{R}^{r\times n}$.

Each column in $W$ is a basis element. A basis element is a component that is repeated several times in the $n$ data points. For example, if our input is a set of faces then ear, nose, eyes, mouth etc. are basis elements which are featured in all facial images. The basis elements are fundamental in reconstructing the original data from approximations. The $H$ matrix aids in reconstructing the approximation is $W$ to the original data points by employing simple linear combination of the basis elements in $W$.

Frobenius norm is used to determine how good the approximation $WH$ is, given as follows,

$$\|X - WH\|_F^2 = \sum_{i,j}(X - WH)_{ij}^2 \tag{4.20}$$

Truncated Singular Value Decomposition (SVD) can be used to obtain an optimal approximation of the Frobenius norm. Finally, we have the following optimization problem for a rank $r$ factorizaton,

$$\min_{W \in \mathbb{R}^{p \times r}, H \in \mathbb{R}^{r \times n}} \|X - WH\|_F^2 \quad \text{such that} \quad W \geq 0 \text{ and } H \geq 0 \tag{4.21}$$

Two-block coordinate descent is the most common NMF algorithm framework used. In this, one of the two factors $W$ or $H$ are optimized first while keeping the other fixed. This techniques is used because an other NP-hard NMF problem can be reduced to a convex problem, more precisely, non-negative least squares problem (NNLS). NNLS can be solved using various techniques and therefore, there are several variants of the NMF algorithm. Some of the popular NNLS approaches include multiplicative updates, alternating least squares, alternating non-negative least squares, and hierarchical alternating least squares [48].

## 4.9 Fast Search and Find of Density Peaks Clustering

Fast Search and Find of Density Peaks Clustering (FSFDPC) [90] is a density and mode based method of clustering. The algorithm draws from the fact that cluster centers are characterized by high density regions which are separated by large distances from each other. Like the mean-shift clustering algorithm [113], the centres of clusters are defined as the local maxima of the density distribution function. However, in the mean-shift method, the data is embedded into a vector space and the density field is maximized for each data point, which is not true for FSFDPC.

We have a data point $i$. Then, we compute the local density $\rho_i$ and distance $\delta_i$ from the points of high density for this data point $i$. These two quantities depend on the distance $d_{ij}$ between data points, which are assumed to satisfy the triangle inequality. The steps of FSFDPC are as follows:

1. Compute the local density $\rho_i$ of a data point $i$ can be written as:

$$\rho_i = \sum_j \chi\left(d_{ij} - d_{\mathrm{c}}\right) \tag{4.22}$$

where $\chi(x) = 1$ if $x < 0$ and $X(x) = 0$ otherwise, and $d_c$ is the cutoff distance.

In short, the local density $\rho_i$ is the total number of points within the radius of the cutoff distance from the data point $i$.

2. Compute the distance $\delta_i$ from the points of high density.

   $\delta_i$ is the minimum of the distance between the point $i$ and any other point of higher density, written as,

$$\delta_i = \min_{j:\rho_j > \rho_i} (d_{ij}) \tag{4.23}$$

   However, for the point with the highest density, the algorithm conventionally uses the following,

$$\delta_i = \max_j (d_{ij}) \tag{4.24}$$

   For the points that are local or global maxima in density, the $\delta_i$ is large compared to the density of their nearest neighbors. This is helpful in finding the cluster centres, i.e. points with large $\delta_i$ values.

3. Identify cluster centres as points with high values of $\rho_i$, and $\delta_i$. Then assign unique labels to each of the cluster centres.

   Now, we can plot a graph where $\delta_i$ is a function of $\rho_i$ and this graph is called a decision graph. The data points with a high $\rho_i$ and $\delta_i$ can clearly be seen as the cluster centres. On the other hand, the data points with high $\delta_i$ but low $\rho_i$ are surrounded by less data points and are isolated from the other clusters. These points are clusters in themselves or can be called outliers.

4. Finally, the unlabelled points are assigned the label of the nearest neighbor of higher density in a single step assignment process.

In Density Based Spatial Clustering of Applications with Noise (DBSCAN) [38], there exists a density threshold and if the density of a point does not meet the threshold then it is assumed to be noise. However, this leads to clusters with low densities being ignored and considered to be noise. In FSFDPC, there is no such noise-signal cutoff. Firstly, the border region for each cluster is found which is defined as the set of points assigned to that cluster but are within the distance $d_c$ from the data points of other clusters. Then the point of highest density is found and its density is denoted by $\rho_b$. Now, $\rho_b$ is used as the threshold where points with densities higher than $\rho_b$ are part of the core of the cluster, otherwise the points form the halo which can also be considered as noise.

## 4.10 Diffusion Learning and Spatial-Spectral Diffusion Learning

Clustering using Diffusion Learning (DL) and Spatial-Spectral Diffusion Learning (DLSS) [80] uses the techniques of graph-based diffusion geometry, and density and mode estimation. Many clustering algorithms use local spatial space parameters, however, DL and DLSS use time of a data-adapted diffusion process scale parameter. The aforementioned property of these two clustering algorithms allows them to cluster data which is multimodal and nonlinear.

### 4.10.1 Diffusion Distance

Diffusion distance is a data adapting measure. When a diffusion process takes place on a graph, it leads to a data-dependent notion of distance which is known as diffusion distance [22]. Diffusion distance has applications in many fields such as molecular dynamics [92], semisupervised learning [28], latent variable separation [64], and data fusion [62]. Diffusion distances can be visualized using diffusion maps. Diffusion maps in turn can be considered as a nonlinear dimensionality reduction method and also aid in computing diffusion distances.

We are provided with $X = \{x_n\}_{n=1}^{N} \subset \mathbb{R}^D$, where $N$ is the number of pixels in the data set and $D$ is the total number of dimensions. Also, $K$ is the number of classes present. The clustering algorithm has to output labels $\{y_n\}_{n=1}^{N}$ where each $y_n \in \{1, \ldots, K\}$.

The underlying geometry of $X$ determines the diffusion distance $d_t(x, y)$ between $x, y \in X$. The time parameter $t$ is used to ascertain the distance as explained below. Let a weighted undirected graph $\mathcal{G}$ encode the geometry of the data $X$. In the graph $\mathcal{G}$, the vertices correspond to $X$ and the edges are determined using the weight matrix $W$ of size $N \times N$, given as follows:

$$W(x,y) := \begin{cases} e^{-\frac{\|x-y\|_2^2}{\sigma^2}}, & x \in NN_k(y) \\ 0, & \text{else} \end{cases} \tag{4.25}$$

where we pick a suitable $\sigma$ and $NN_k(y)$ is the set of k-nearest neighbors of $y$ in $X$ using Euclidean distance.

Then, the weight matrix $W$ is normalized to be row stochastic that yeilds a Markov diffusion with transition matrix $P$ given as follows:

$$P(x,y) = \frac{W(x,y)}{\sum_{z \in X} W(x,z)} \tag{4.26}$$

Now, we have an initial distribution $\mu \in \mathbb{R}^N$ on the state space. Using the transition matrix, the next state of the Markov chain at time $t \geq 0$ is the vector $\mu P^t$. The diffusion process on $X$ evolves according to the connections between the points as the time $t$ increases. The Markov chain has a stationary distribution $\pi$ s.t. $\pi P = \pi$, given by,

$$\pi(x) = \frac{\deg(x)}{\sum_{y \in X} \deg(y)} \tag{4.27}$$

$$\deg(y) = \sum_{x \in X} P(x,y) \tag{4.28}$$

Now, the diffusion distance at time $t$ can be written as,

$$d_t^2(x,y) := \sum_{u \in X} \left(P^t(x,u) - P^t(y,u)\right)^2 d\mu(u)/\pi(u) \tag{4.29}$$

$d_t(x,y)$ is computed by summing over all paths of length $t$ that connect $x$ to $y$. Therefore, the diffusion distance is small if $x$ and $y$ are strongly connected according to the transition matrix and vice versa.

We can compute the diffusion distance $d_t$ faster by employing eigen decomposition of $P$ matrix. Under mild conditions, the matrix $P$ admits a spectral decomposition of eigenvectors $\{\Phi_n\}_{n=1}^N$ and eigenvalues $\{\lambda_n\}_{n=1}^N$, where $1 = \lambda_1 \geq |\lambda_2| \geq ... \geq |\lambda_N|$. The diffusion distance in terms of the above mentioned spectal decomposition is as given below,

$$d_t(x,y) = \sqrt{\sum_{n=1}^{N} \lambda_n^{2t}(\Phi_n(x) - \Phi_n(y))^2} \tag{4.30}$$

The time parameter $t$ decides for how long the diffusion process on the graph $\mathcal{G}$ takes place which in turn decides the diffusion distance. A smaller value of $t$ means less diffusion which means that the diffusion distance is small. This prevents the discovery of interesting geometric information present in the data but however, the small information of the geometry that we do have is very detailed. If the value of $t$ is too long then the interesting geometric information is washed away. There should be a balance which is when the geometry is revealed. This is achieved at $t = 30$.

### 4.10.2 Clustering Algorithm

The data set $X$ is reshaped into an $N \times D$ matrix, where $N$ is the number of pixels in the image and $D$ is the number of spectral bands. We then consider the image $X$ to be a collection of points $\{x_n\}_{n=1}^{N} \subset \mathbb{R}^D$.

The algorithm consists of two parts, which are,

- Mode Identification
- Labeling of Points

#### Mode Identification

The steps for the first part of the clustering algorithm are as follows:

1. Compute the empirical densities $\{p(x_n)\}_{n=1}^{N}$ for all the elements of $X$. Using a kernel density estimator, for each $n \in \{1, ..., N\}$,

$$p_0(x_n) = \sum_{x_m \in NN_k(x_n)} e^{\frac{-||x_n - x_m||_2^2}{\sigma_2}} \tag{4.31}$$

where $||x_n - x_m||_2^2$ is the squared Euclidean distance in $\mathbb{R}^D$ and $NN_k(x_n)$ is the set of $k$-nearest neighbors to $x_n$ in Euclidean distance. The empirical density $p$ is calculated by normalizing the density $p_0$ so that we have $\sum_{n=1}^{N} p(x_n) = 1$, as given,

$$p\left(x_n\right) = p_0\left(x_n\right) / \sum_{m=1}^{N} p_0\left(x_m\right) \tag{4.32}$$

2. Compute $\{\rho_t(x_n)\}_{n=1}^{N}$. $\rho_t$ is a time dependent quantity. It assigns to each pixel the minimum diffusion distance between the pixel and a point of higher empirical density. The point with the highest empirical density is assigned the maximum diffusion distance between it and any other point as its $\rho_t$ value.

   $$\tilde{\rho}_t\left(x_n\right) = \begin{cases} \min_{\{p(x_m)\geq p(x_n)\}} d_t\left(x_n, x_m\right), & x_n \neq \arg\max_i p\left(x_i\right) \\ \max_{x_m} d_t\left(x_n, x_m\right), & x_n = \arg\max_i p\left(x_i\right) \end{cases} \tag{4.33}$$

   where $d_t$ is the diffusion distance betweeen $x_m$ and $x_n$ at time $t$. From here on, we use the normalized version $\rho_t\left(x_n\right) = \tilde{\rho}_t\left(x_n\right) / \max_{x_m} \tilde{\rho}_t\left(x_m\right)$, therefore therefore the maximum of $\rho_t\left(x_n\right)$ is 1.

3. The modes $x_1^*, ..., x_K^*$, where $K$ is the number of clusters, are the points which yield the $K$ highest values of the following quantity,

   $$\mathcal{D}_t\left(x_n\right) = p\left(x_n\right) \rho_t\left(x_n\right) \tag{4.34}$$

   The above formula ensures that the modes are points with high density and far in diffusion distance from other higher density points. Therefore, these points are considered to be the modes of different distributions. However, this method to find modes in the data is accurate under the assumption that the data is drawn from nonparametric distributions [73].

### 4.10.3 Labeling Points

Each mode after the mode identification process is assigned a unique label. Consequently, the rest of the sample points are given labels based on the following process.

The labeling of points process for assigning labels to the rest of the points:

- Firstly, the points are sorted in the order of decreasing empirical density. In the order of decreasing empirical density, we compute the *spatial consensus* *label* for each of the leftover points. This is done by considering all the labeled points in the spatial radius of $rs \geq 0$ which are the $NN^s_{r_s}$ set of points. Amongst these $NN^s_{r_s}$ points, if a point occurs with frequency $>$ 0.5, then the label of that point is the spatial consensus label. In that case that the above is not true, then there is no spatial consesus label.

Let $L_n^{\text{spatial}} = \left\{ y_m | x_m \in NN^s_{r_x}\left(x_n\right), x_m \neq x_n \right\}$ be the spatial neighbors in the radius $r_s$ of a given point $x_i$ in consideration. Then the spatial consesus label o0f $x_i$ can be written as:

$$y_i^{\text{spatial}} = \begin{cases} k, & \frac{1\left\{ y_n \middle| y_n = k, y_n \in L_n^{\text{revinil}} \ y \right|}{\left| L_n^{\text{fillin}} \ \right|} > \frac{1}{2} \\ 0(\text{ no label }), & \text{else.} \end{cases} \tag{4.35}$$

- Once the spatial consensus label of a point is computed, we go further to assign the spectral label. The spectral label is the nearest neighbor in the spectral domain (measured in diffusion distance) and is of higher density compared to $x_i$.
- $x_i$ is then given the final label. The final label is the spectral label unless the spatial consensus label exists and is different from the spectral label. If the spatial consensus label exists and is different from the spectral label then that $x_i$ is not assigned any label in the first stage. In the case where a point is unlabeled, it is assigned the label 0 in order to efficiently compute the spatial consensus label. If the $L_{nspatial}$ mostly has unlabeled points then the spatial consensus label for that $x_i$ is 0.
- After the above steps, the data is partially labeled. In the final step, the unlabeled points are assigned the final label which is same as the spatial consensus label if it exists. If not, then the label is the label of the nearest spectral neighbor of higher density.

Now, the points with a high density are mostly labeled in accordance with their spectral properties. This is because these points are more likely to be closer to the centres of distribution, which provide an overall spatially homogeneous region. Secondly, high density points are labeled before the low density points and at that stage most points around the high density point are unlabeled, which means the spatial consensus label does not exist. On the contrary, the low density points are scattered more towards the edges of clusters or distributions. This is why they are more likely to have labels according to their spatial properties. DLSS successfully employs both spectral and spatial information.

The difference between DL and DLSS lies in the labeling of points process. In DL, all the unlabeled points are assigned the label of the nearest neighbor of higher density. In this way, the authors have neatly compared DL and DLSS and empirically proved that DLSS that employs spatial information fairs better than DL.

# Chapter 5

# Theoretical Comparison of Methods

In this chapter, we provide information on how to differentiate between various algorithms considered in this work based on theoretical and algorithmic distinctions. Firstly, we look at the key conclusions drawn from the analysis, then we delve into how to categorize clustering techniques using different criteria and finally we explain how we came to the conclusions we made in the first subsection.

## 5.1 Key Points from the Analysis

In our experiments, we look at a variety of methods for clustering hyperspectral images. We arrive at several conclusions after a thorough analysis which are explained in detail later and briefly described as follows:

- Classical clustering techniques have unreliable discriminative abilities.
- Dimensionality reduction techniques are used in conjunction with classical clustering techniques but also suffer from unreliable discriminative abilities.
- Deep learning methods are more computationally efficient techniques that can be applied to high-dimensional and high-semantic data. However, they often fail to find the inherent pattern in several datasets without some supervision and domain knowledge.

- Manifold clustering techniques are computationally less efficient, have longer run times, and lack scalability. They also have limited representation power.
- Non-negative matrix factorization techniques are computationally efficient.
- Out of the three density based techniques: FSFDPC, DL, and DLSS, DLSS performs the best in terms of speed and accuracy.

## 5.2 How to categorize clustering techniques?

There are several ways to categorize clustering techniques as depicted in Table 5.1.

Table 5.1: This table depicts ways to categorize the clustering techniques based on the various listed criteria.

| Algorithms | Clustering Category | Deep Technique | Dimension Reduction | Distance Metric | Time Complexity |
|---|---|---|---|---|---|
| k-means | Partitional | ✗ | ✗ | Euclidean | $\mathcal{O}(n)$ |
| PCA + KM | Partitional | ✗ | ✓ | Euclidean | $\mathcal{O}(n)$ |
| Auto + KM | Partitional | ✓ | ✓ | Euclidean | $\mathcal{O}(n)$ |
| DEC | Partitional | ✓ | ✓ | Euclidean | $\mathcal{O}(n)$ |
| GMM | Partitional | ✗ | ✓ | Euclidean | $\mathcal{O}(n)$ |
| SMCE | Partitional | ✗ | ✓ | Euclidean | $\mathcal{O}(i)$ |
| HNMF | Hierarchical | ✗ | ✗ | Euclidean | $\mathcal{O}(n)$ |
| FSFDPC | Partitional | ✗ | ✗ | Euclidean | $\mathcal{O}(n^2)$ |
| DL | Partitional | ✗ | ✓ | Diffusion | $\mathcal{O}(nlogn)$ |
| DLSS | Partitional | ✗ | ✓ | Diffusion | $\mathcal{O}(nlogn)$ |

Clustering techniques can be divided into partitional or hierarchical techniques. The basic distinction is made in terms of whether the clustering is nested or unnested. In partitional clustering, we divide the dataset into non-overlapping clusters whereas in hierarchical clustering the clusters are nested clusters and organized in the form of a tree. Hierarchical clustering does not assume the number of clusters $k$ and produces a more interpretable and meaningful taxonomy while clustering. It also uses only proximity metric or a distance metric to form new clusters. Hierachical clustering can further be of two types i.e. agglomerative and divisive. Agglomerative is a approach where data points are merged together based on some similarity metric to form initial clusters. In the next steps, these clusters are merged together and the process is continued until there are no more individual data points left. Divisive clustering is a top-down approach where all points are initially part of one cluster which is divided repeatedly to have only singleton clusters of individual data points. All algorithms considered in this work fall under the category of partitional clustering except for HNMF which falls under hierarchical clustering.

Another approach to comparing the algorithms mentioned in this work is by looking at whether or not the algorithms are deep techniques. Classical techniques like $k$-means clustering use the notion of distance to find similarities in the input data which is why they are shallow models. Shallow models fail to find discriminative properties and semantic similarities in the given data. Deep learning models have several advantages over shallow models. As mentioned in [79], the use of the mini-batch Stochastic Gradient Descent (SGD) for propagating weights in a neural network makes deep learning models computationally more efficient, they can project data into lower-dimensional spaces very easily, and they scale well for high-dimensional and large-scale datasets due to their multi-layer architecture. In this work, autoencoders are the deep learning models used for clustering. We consider several different variations of autoencoders based on reconstruction loss, followed by k-means clustering to cluster the hyperspectral data. Secondly, we consider the algorithm DEC as mentioned in section 4.6. However, deep learning techniques do not answer all our problems. They lack robustness and need extensive hyperparameter tuning. It is also difficult to learn the hidden pattern in the given input datasets without some supervision and domain knowledge about the dataset.

Section 4.10.1 delves into diffusion distance and its advantages over Euclidean distance. The techniques used in this work can be divided into categories that use diffusion distance and the ones that use Euclidean distance. This is clearly shown in Table 5.1.

Lastly, we analyse the time complexities of the algorithms (last column of Table 5.1). The constants mentioned in the table imply the following, number of samples, $n$, and number of iterations performed for optimization, $i$. The constants for computing time complexities for DL and DLSS are mentioned in the original work [80] in detail. The time complexities of k-means clustering, PCA followed by k-means clustering and HNMF have a time complexity of $\mathcal{O}(n)$, where $n$ is the number of samples, therefore they take the least amount of time to cluster datasets. This is empirically shown in later Section 6 in Table 6.3. The density-based technique FSFDPC as well as SMCE have some of worst time complexities and have a longer run time. Also, the deep learning techniques need to be pre-trained which adds to the run time, however, their clustering time complexity is low $\mathcal{O}(n)$.

## 5.3 Comparative Analysis of Clustering Techniques

Firstly, we consider $k$-means clustering and give an HSI as an input. One of the major drawbacks of $k$-means clustering is that it assumes that the clusters are present in a spherical shape. HSI datasets are have a non-linear inherent pattern which $k$-means also fails to learn. $k$-means clustering can also converge to a local minima and is highly dependent on the initial centroid assignments. This is why we move on to more theoretically advanced techniques and scrutinize them for their advantages and disadvantages.

HSIs are high-dimensional images and the curse of dimensionality can be tackled by using some of the dimensionality reduction techniques in conjunction with k-means clustering. This is seen in methods mentioned in sections 4.4 and 4.6. Section 4.4 explains that PCA is a linear technique and an autoencoder with non-linear activation functions can perform non-linear dimentionality reduction. Theoretically, the non-linear dimensionality reduction techniques should capture more information that aids our next clustering step. However, we see in later sections, during empirical analysis of these techniques, that they fail to project the original to a low-dimensional space where we achieve high class separability. These techniques merely help in reducing the high-dimensional dataset to a low-dimensional latent space representation and have a low representation power.

Methods like $k$-means and Gaussian Mixture models (GMM) make assumptions about the data and data distribution. However, all datasets do not satisfy the geometric and shape requirements of these methods and therefore these methods perform poorly. Spectral clustering techniques like Sparse Manifold Clustering and Embedding (SMCE) analyse a matrix constructed based on point-to-point similarities using Euclidean distance and work better. However, to find the eigenvalues of the affinity matrix may take longer computation time which is a major drawback when applied to high dimensional HSIs.

SMCE can also be compared to Spatial-spectral diffusion learning (DLSS) and diffusion learning (DL) algorithms. SMCE algorithm employs a graph Laplacian to compute the eigenvectors and uses nonlinear distances. In DLSS and DL, the authors in [80] compute the eigenvectors of the Markov transition matrix to build the diffusion maps. The major difference is in the method employed for clustering after computing the eigenvectors. SMCE uses k-means clustering whereas DLSS and DL use a mode based estimation technique. Also, in SMCE as the name suggests, the algorithm has sparsity assumptions and employs a sparse optimization solution which DLSS and DL do not. SMCE is not robust and also does not scale well for large datasets. Therefore, DL and DLSS clearly are an improvement over SMCE.

Nonnegative Matrix Factorization (NMF) and Hierarchical Nonnegative Matrix Factorization (HNMF) both have sparsity constraints. However, DL and DLSS do not which suit better for our objective and application to hyperspectral datasets.

FSFDPC, DL and DLSS use mode estimation techniques and employ density based analysis to find modes of the clusters. However, FSFDPC differs from DL and DLSS as it employs Euclidean distance to find the distances between cluster centres whereas DL and DLSS use diffusion distance. As diffusion distance is able to use the geometric information contained in the data, it is more efficient in finding modes when used by DL and DLSS. The algorithms also differ by how the labels are assigned to points after the discovery of the modes. In DL and DLSS, the spectral neighbors are used which are the nearest neighbors found using diffusion distance. Then the unlabeled points are assigned the label of its spectral neighbor of the highest density. In FSFDPC, nearest neighbors are employed where Euclidean distance is used to find the neighbors. Also, DLSS employs spatial information for further assigning labels to unlabelled points which is not seen in DL and FSFDPC. Therefore, DLSS performs better that FSFDPC and DL.

The conclusion of this analysis is that DLSS, despite algorithmic complextiy, $\mathcal{O}(nlogn)$, is likely to best cluster hyperspectral datasets. This will be analyzed experimentally in Chapter 6.

# Chapter 6

# Experimental Comparison of Methods

In this chapter, we experimentally compare the methods when applied to hyperspectral datasets. We first discuss the evaluation metrics used to test the correctness of the experiments. We study and analyse the performance of different clustering techniques on the basis of the above mentioned evaluation metrics. Then, we evaluate the performance of different autoencoders when implemented on hyperspectral data. Finally, we look at how changes to the hyperparameters, mainly, learning rate and latent space dimension size affect the clustering performance.

## 6.1 Evaluation Metrics

Specific evaluation metrics are used in this work to analyze the performance of the dimensionality reduction and clustering techniques numerically. We measure overall accuracy (OA), average accuracy (AA), Fisher's discriminant ratio and run time.

### Overall Accuracy (OA)

We use overall accuracy to understand the final clustering result to learn what percentage of the samples are correctly clustered. This is done by comparing the predicted cluster labels with the ground truth labels.

Overall accuracy is computed by dividing the number of correct classifications by the total number of samples in the dataset. A 100% overall accuracy means every pixel is classified correctly.

$$OA = \frac{\text{Number of correctly predicted samples}}{\text{Total number of samples}} \tag{6.1}$$

### Average Accuracy (AA)

Another evaluation metric to study the final clustering result is average accuracy. This helps to identify how the clustering technique performs for each individual class and the final metric is the mean over the classes.

Average accuracy is the average taken over all the class accuracies. This is used primarily to take into account class imbalance and weights small and large classes equally. Let there be $k$ total classes in the dataset. Let us compute the class accuracy (CA) for each class $i$ is :

$$CA_i = \frac{\text{Number of correctly predicted samples of class } i}{\text{Total number of samples in class } i} \tag{6.2}$$

Now, average accuracy (AA) can be written as:

$$AA = \frac{\sum_{i=1}^{k} CA_i}{k} \tag{6.3}$$

### Fisher's discriminant ratio

We use Fisher's discriminant ratio to quantitatively understand the effect of dimensionality reduction. It is a measure of class separability in feature space. It is invariant to the scale of data samples and also to the number of dimensions $d$ in a particular feature space. This enables us to employ the metric to multiple datasets and algorithms for a consistent comparative analysis.

Fisher's discriminant ratio is computed for a pair of classes, as the ratio of the between-class scatter and the within-class scatter.

Let us say that we have a feature space with data samples of $d$ dimensions in class $A$ and class $B$, with means $\mu_A$ and $\mu_B$ respectively, the Fisher's discriminant ratio can be computed as follows:

$$J(A,B) = \frac{\left\|\mu_A - \mu_B\right\|^2}{S_A^2 + S_B^2} \tag{6.4}$$

where $J$ is the Fisher's discriminant ratio for the pair of classes, $\|.\|_2$ is the $L_2$ norm, and $S_i^2$ is the within-class scatter of a specific class $i$. $S_i^2$ is as:

$$S_i^2 = \frac{1}{N_i} \sum_{n \in N_i} \left\|x_n - \mu_i\right\|^2 \tag{6.5}$$

where $x_n$ is a point in class $i$ with $N_i$ total points.

For a good feature space representation, we obtain a high Fisher's discriminant ratio value when the means of the pair of classes are farther apart than the points within a class are closer to each other.

### Time

We use time to understand the computational efficiency of the methods considered in this work. It is computed by summing the time taken by each process in the clustering method.

## 6.2 Performance of clustering methods applied to hyperspectral data

### 6.2.1 Experimental Setup

We conduct our experiments on three datasets: Salinas-A, Pavia and Indian Pines as described in Chapter 2. In [80], the authors have restricted the spatial resolution of the Pavia and Indian Pines datasets in order to reduce the number of classes and form well separated clusters. Authors in [116] prove that the computational complexity of their model grows exponentially with increase in the number of clusters and when there are more than 10 clusters present the clustering performance deteriorates. We empirically analyse these guarantees for various clustering techniques when applied to hyperspectral datasets. In this work, we keep the Indian Pines dataset as is and do not change the spatial resolution or the number of classes. Also, to be able to better compare results with that in [80], we prune the Pavia dataset to have 6 instead of 9 total classes.

Some clustering techniques cannot predict the number of clusters to be estimated and for those techniques we provide the number of clusters present in the ground truth (GT) to be the number of clusters to be estimated. This number is denoted by $k_T$ throughout this work.

As we are analyzing unsupervised techniques, we input the entire dataset along with the points with no or unknown ground truth labels. However, we do not include points with no or unknown class in the ground truth during our evaluation.

Most of the parameters and hyperparameters employed are same as in the original works ([80], [109], [108]) to allow better comparative analysis. Further, we do a thorough analysis of the hyperparameters: latent space dimension size and learning rate for the dimensionality reduction and deep learning techniques in later sections.

### 6.2.2 Analysis

We find that DLSS outperforms the other methods for the all datasets as in Table 6.1. DEC, SMCE, HNMF, and FSFDPC perform equally well but we notice aberrations for various datasets.

Table 6.1: Comparison of overall clustering accuracy in percentages for each algorithm implemented on different hyperspectral datasets

| Datasets | Number of Classes | Clustering Accuracy (OA) (in percentages) | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | k-means | PCA + k-means | Auto + k-means | DEC | GMM | SMCE | HNMF | FSFDPC | DL | DLSS |
| Salinas | 6 | 62.5 | 62.5 | 30.70 | 70.96 | 76.80 | 46.62 | 63.20 | 63.22 | 83.13 | **84.76** |
| Pavia | 6 | 77.6 | 77.55 | 79.24 | 72.17 | 85.38 | 83.52 | 72.17 | 77.83 | 84.9 | **93.6** |
| Indian Pines | 16 | 39.6 | 39.42 | 34.45 | 38.76 | 38.89 | 33.89 | 36.36 | 39.16 | 35.78 | **41.82** |

Authors in [117] observe and prove that the overall accuracy decreases as the number of clusters increase. This can be observed in the low overall accuracies (OA) for all methods when considering the Indian Pines datasets.

As noted in [10] for RGB datasets, DEC employs a feed-forward artificial neural network instead of a convolutional neural network. Due to this, DEC can not capture local information as well as DLSS which clearly utilizes spatial information in its labelling scheme.

A major drawback of SMCE can be observed from Table 6.3. SMCE takes longer to assign clusters to samples than DLSS. The same is also noted for DEC which takes longer than DLSS on account of being a deep learning technique. Also, the linear variants of $k$-means clustering i.e. PCA with $k$-means clustering take lesser computational time and memory compared to the deep and nonlinear variants i.e. autoencoder with $k$-means clustering and DEC.

Given that the hyperspectral datasets are nonlinear, a fair assumption would be to say that the deep and nonlinear variants of $k$-mean clustering perform better than the linear variants. In the results in Table 6.1, we notice this for DEC. However, there is a drop in performance for autoencoder being employed along with $k$-means clustering. This is due to the fact that DEC employs a clustering loss along with reconstruction loss, whereas, the autoencoder is first trained to obtain a latent space representation and then the $k$-means clustering is applied to the latent space. Simulataneously reconstructing and clustering helps to improve the latent space representational power of autoencoder which is seen from the results in DEC.

Figure 6.1 shows the final clustering result for the various algorithms. We can compare the clustering result of each algorithm with the ground truth labels. We observe that the deep learning techniques like autoencoder along with $k$-means and DEC have high overall accuracies and comparatively low average accuracies from Tables 6.1 and 6.2. From Figure 6.1, specifically from subfigure (e), we can understand that this is due to the fact that final class labels are more scattered, so even if the algorithm successfully predicts more points correctly, it fails to do so for each class. Another conclusion that can be drawn from this is that the multi-layer perceptron and 1-D convolutional deep learning techniques considered fail to capture the information in the neighborhood of each pixel. Similar pixels are present close to each other which calls for techniques that allow neighborhood information and spatial information preservance. This vital difference is beautifully depicted in Figure 6.1 by DL (subfigure (j)) and DLSS (subfigure (k)) algorithms where DL only considers spectral information and performs worse than DLSS which also uses spatial information. We see that the green class is correctly and completely identified by DLSS where as in DL only part of the green class is correctly identified. The spatial labelling scheme used by

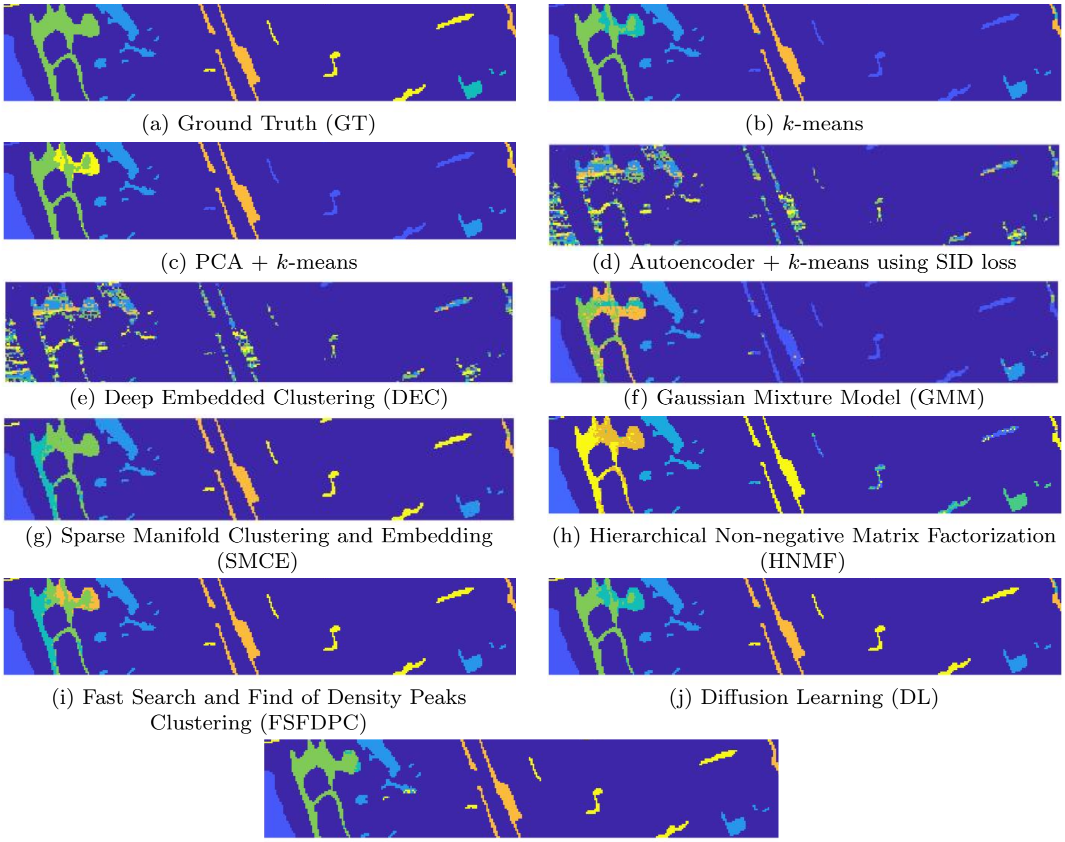
DLSS helps it to propagate the spectral information gained from the mode detection scheme to the entire green class.

Table 6.2: Comparison of average clustering accuracy in percentages for each algorithm implemented on different hyperspectral datasets

| Datasets | Number of Classes | Clustering Accuracy (AA) (in percentages) | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | *k*-means | PCA + *k*-means | Auto + *k*-means | DEC | GMM | SMCE | HNMF | FSFDPC | DL | DLSS |
| Salinas | 6 | 65.77 | 65.77 | 28.92 | 69.28 | 74.20 | 42.01 | 66.42 | 60.55 | 87.9 | **89.76** |
| Pavia | 6 | 62.39 | 62.37 | 76.08 | 66.44 | 41.20 | 77.15 | 74.22 | 74.65 | 77.87 | **82.10** |
| Indian Pines | 16 | **42.02** | 37.33 | 40.87 | 27.88 | 30.53 | 31.56 | 35.09 | 35.12 | 29.94 | 33.57 |

Table 6.3: Comparison of run time in seconds for each algorithm implemented on different hyperspectral datasets

| Datasets | Time (in seconds) | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | *k*-means | PCA + *k*-means | Auto + *k*-means | DEC | GMM | SMCE | HNMF | FSFDPC | DL | DLSS |
| Salinas | 0.69 | **0.01** | 16.37 | 31.34 | 8.05 | 180.86 | 0.45 | 3.42 | 4.44 | 6.11 |
| Pavia | 2.71 | **0.01** | 50.01 | 69.51 | 6.95 | 313.60 | 0.53 | 10.74 | 14.76 | 30.69 |
| Indian Pines | 27.59 | **0.01** | 15.00 | 111.32 | 64.18 | 270.56 | 1.29 | 28.79 | 49.84 | 41.82 |



(a) Ground Truth (GT)

(b) $k$-means

(c) PCA + $k$-means

(d) Autoencoder + $k$-means using SID loss

(e) Deep Embedded Clustering (DEC)

(f) Gaussian Mixture Model (GMM)

(g) Sparse Manifold Clustering and Embedding (SMCE)

(h) Hierarchical Non-negative Matrix Factorization (HNMF)

(i) Fast Search and Find of Density Peaks Clustering (FSFDPC)

(j) Diffusion Learning (DL)

(k) Spectral-Spatial Diffusion Learning (DLSS)

Figure 6.1: Clustering results for the following techniques: (a) Ground Truth, (b) $k$-means, (c) PCA + $k$-means, (d) Autoencoder with SID loss + $k$-means, (e) DEC, (f) GMM, (g) SMCE, (h) HNMF, (i) FSFDPC, (j) DL, and (k) DLSS for the Pavia Centre dataset. We see that DLSS outperforms all other techniques. DL performs second best. DLSS is better than DL as it employs a labelling scheme that employs spatial information. SMCE which is a manifold clustering technique and has the forth best OA (after GMM) but the AA is comparable to that of DL. SMCE is able to preserve neighborhood information in the manifolds where clustering takes place. GMM has the third best OA but the worst AA which can be clearly seen in the figure (f). It misclassifies most of the smaller classes. We also notice that after SMCE, autoencoders have high OA and AA but from figure (d), we see that it is does not produce spatially smooth results. There is a lot of salt and pepper noise in the clustering result.

## 6.3 Performance of Autoencoders on Hyperspectral Data

In this section, we compare the performance of different autoencoders implemented with hyperspectral data as the input. The evaluation metrics used are overall accuracy and average accuracy. The results in this section are for a latent space dimension of size 2 which is determined using a thorough analysis as discussed in Section 6.5.

### 6.3.1 Analysis

In Table 6.4, the multi-layer perceptron autoencoders with sum of squared error (SSE), cosine of spectral angle (CSA) and spectral angle (SA) as loss functions, perform equally well on the Pavia dataset. We see an improvement in the performance with the use of spectral information divergence (SID) in the loss function. Finally, the 1-D convolutional autoencoder performs the second best with an overall accuracy of 77.45 % and average accuracy of 75.48 % on the Pavia dataset.

We also look at the final clustering results in Figure 6.2. We notice a lot of salt and pepper noise (subfigures (b) to (f)) in the results. The autoencoders tested so far successfully recover the spectral information but are spatially inconsistent. We know that neighboring samples come from the same cluster which the autoencoders are not able to pick up on. A 3-D convolutional autoencoder would theoretically perform better in this regard which we will look at in the future.

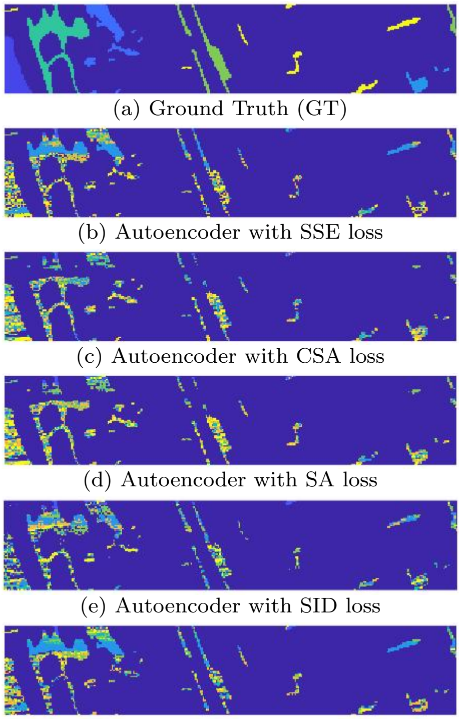
Figure 6.3 uses the best two latent space dimensions to display the obtained feature space by each autoencoder in a 2-D space. The standard deviation in the direction of a specific latent space dimension is produced and sorted in the ascending order to find the best two latent space dimensions to represent the feature space. We do not notice a major difference between any of the latent representations. We do notice that there is high variance in the clusters which makes it harder for algorithms like $k$-means clustering to perform well. This is due to the assumption made by $k$-means clustering that all clusters are present as spheres instead of ellipsoids.

Table 6.4: Comparison of overall clustering accuracy in percentages for different autoencoders implemented on different hyperspectral datasets for latent space dimension size of 2.

| Datasets | Number of Classes | Clustering Accuracy (OA) (in percentages) | | | | |
|---|---|---|---|---|---|---|
| | | SSE | CSA | SA | SID | Convolutional |
| Pavia | 6 | 70.14 | 66.36 | 67.94 | **77.45** | 70.89 |

Table 6.5: Comparison of average clustering accuracy in percentages for different autoencoders implemented on different hyperspectral datasets for latent space dimension size of 2

| Datasets | Number of Classes | Clustering Accuracy (AA) (in percentages) | | | | |
|---|---|---|---|---|---|---|
| | | SSE | CSA | SA | SID | Convolutional |
| Pavia | 6 | 64.21 | 60.61 | 39.68 | **75.48** | 69.09 |



(a) Ground Truth (GT)

(b) Autoencoder with SSE loss

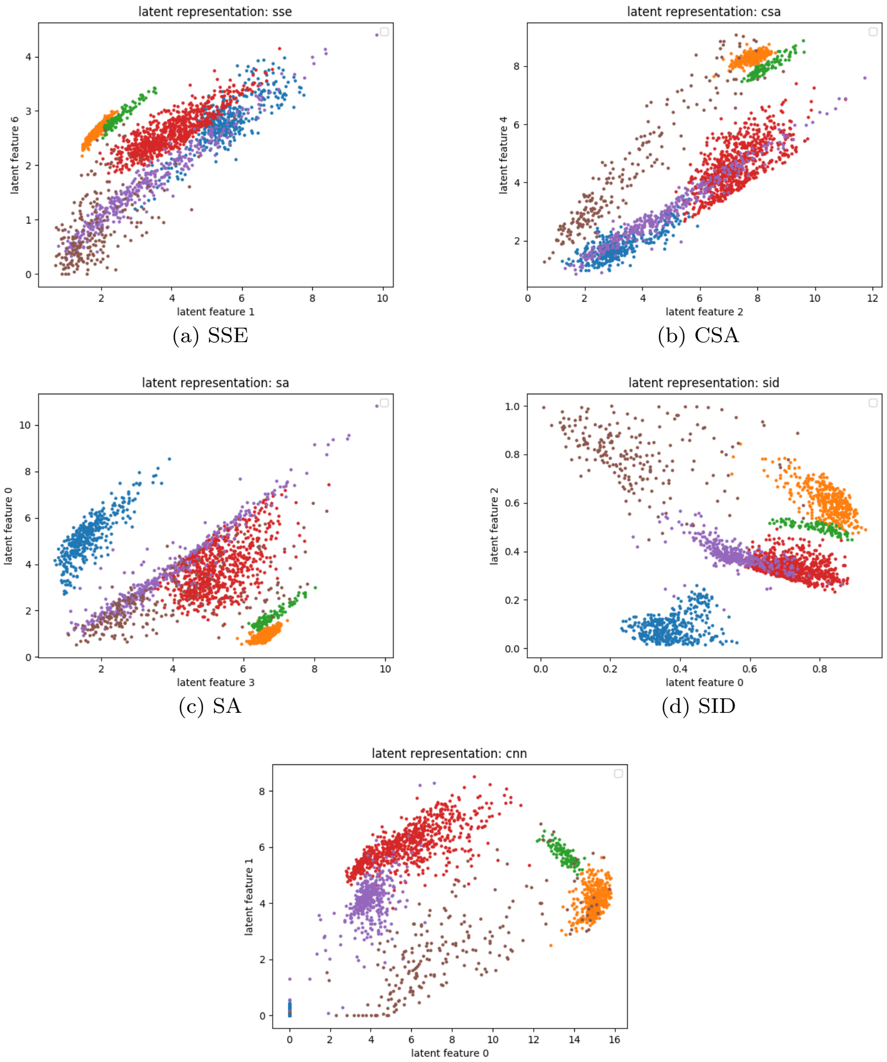
(c) Autoencoder with CSA loss

(d) Autoencoder with SA loss

(e) Autoencoder with SID loss

(f) Autoencoder with CNN loss

Figure 6.2: Clustering results of different autoencoders on the Pavia Centre dataset. The figures represent the following algorithms: (a) Ground Truth, (b) Autoencoder with SSE loss, (c) CSA loss, (d) SA loss, (e) SID loss, (f) CNN. The autoencoder with SID loss has the highest OA and AA values which can be (seen in figure (e)), followed by CNN (figure (f)) and SSE (figure (b)). We also notice that autoencoder produce spatially less smooth clustering results. We theorize that this is due to the fact that autoencoders do not have a mechanism to preserve neighborhood information for every pixel.



(a) SSE

(b) CSA

(c) SA

(d) SID

(e) CNN

Figure 6.3: Best latent dimension representation for the Pavia University dataset for various autoencoders (a) SSE, (b) CSA, (c) SA, (d) SID, and (e) CNN. The standard deviation in the direction of a specific latent space dimension is produced and sorted in the ascending order to find the best two latent space dimensions to represent the feature space. We do not notice a huge difference between the latent dimension representation for the different autoencoders considered in this work. We do see that there is high variance and that the clusters are present as ellipses.

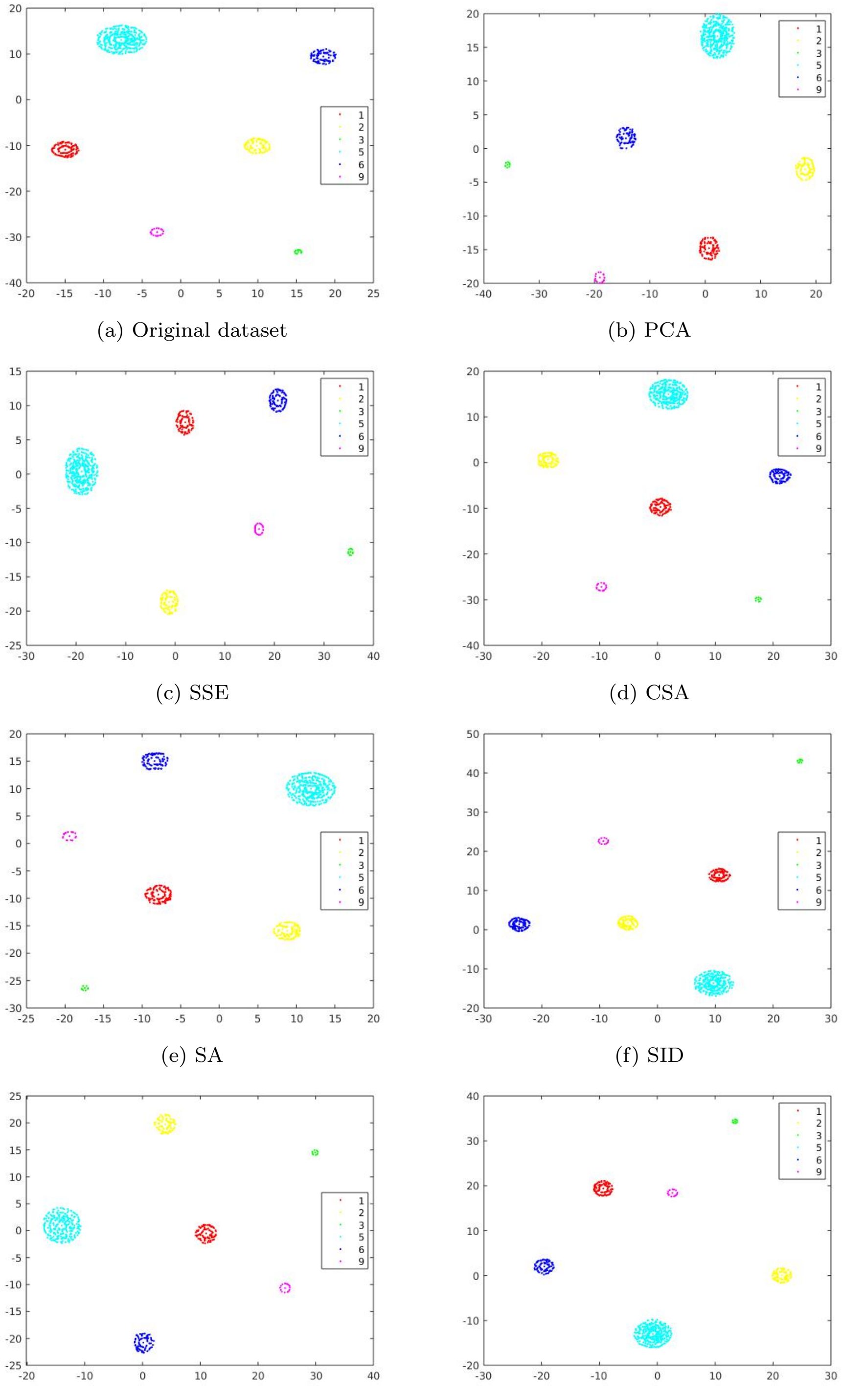
## 6.4 Feature Space Analysis

### t-Distributed Stochastic Neighbor Embedding

t-Distributed Stochastic Neighbor Embedding (t-SNE) [104] is a nonlinear dimensionality reduction technique which is most popularly used to visualize high-dimensional datasets. It is used in image processing [17, 104], bioinformatics and computational Biology [69], natural language processing (NLP) [17], etc. Once t-SNE is applied to a dataset, the input features in the output of t-SNE are not recognizable. This is the reason why t-SNE is primarily used for data familiarization, exploration and visualization.

t-SNE is different from the dimensionality reduction techniques referred to in Section 4.2 and 4.3 because these dimensionality reduction techniques solve different minimization problems. t-SNE focuses on preserving the local distances and neighborhood patterns between data points in the low-dimensional space.

In Figure 6.4, we show the t-SNE representations for the various latent spaces of the dimensionality reduction technqiues considered in this work. We can not draw conclusion based entirely on these figures as there is no noticeable difference. Therefore, we employ Fisher's discriminant ratio as in Section 6.5.2 to better evaluate these techniques. We can safely conclude that DEC has the highest latent space dimension representational power for clustering datasets. However, PCA comes second to DEC and also has a much lower computational time. Another fact to consider is that none of the techniques considered so far use spatial information which can aid clustering performance. Therefore, we would like to experiment with techniques that employ spatial information such as a 3-D convolutional autoencoder.



(a) Original dataset

(b) PCA

(c) SSE

(d) CSA

(e) SA

(f) SID

(g) CNN

(h) DEC

Figure 6.4: t-SNE representations for the Pavia University dataset for various dimensionality reduction techniques (a) Original Dataset, (b) PCA, autoencoders (c) SSE, (d) CSA, (e) SA, (f) SID, (g) CNN, and (h) DEC

## 6.5 Effect of Learning Rate and Latent Space Dimension Size on Autoencoders

In this section, we look at how changes made to the hyperparameters i.e. learning rate and latent space dimension size affect the clustering result. We first consider the learning rate for the autoencoders. Then we look at the size of the feature space for various algorithms.

### 6.5.1 Autoencoders: Learning Rate

Learning rate plays a crucial role in deep networks as it affects whether or not the network converges to a global optimum. We study the effect of learning rate on the clustering result for the SID autoencoder with latent space dimesion of size 10. We do so by looking at the changes observed in OA as the learning rate is changed. We see that there is a wide variance in the overall accuracy as the learning rate is varied. The best accuracies are obtained for a learning rate of $10^{-3}$ which are 58.48% OA and 64.82% AA. This learning rate is consistently used for all deep learning algorithms considered in this work. We obtain 34.08% OA and 0% AA for $10^{-1}$, 23.45% OA and 11.82% AA for $10^{-1}$, and 51.28% OA and 52.17% AA for $10^{-4}$.

### 6.5.2 Dimensionality Reduction Techniques and the Size of Latent Space

We consider the size of the latent space for hyperparameter analysis of dimensionality reduction technqiues. We do so first by considering the dimensions from $r = 1$ to $r = 10$ for the SID autoencoder. Figure 6.5 shows the plot of overall accuracy and average accuracy for the various latent space dimension sizes. We observe that $r = 2$ and $r = 3$ produce comparable performance. Therefore, we further analyse these two latent space dimension sizes. We also compare the results with $r = 10$ as these have been considered in [109] and [108].

We look at the Fisher's discriminant ratio, overall accuracy and average accuracy for various dimensionality reduction algorithms, and the results are shown in Figures 6.6, 6.7 and 6.8.

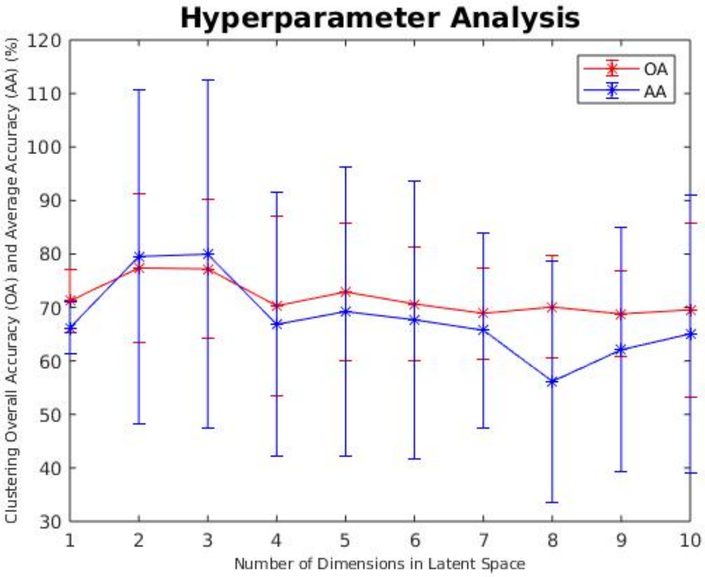


Figure 6.5: Hyperparameter analysis based on different number of latent space dimensions using overall and average accuracy for SID autoencoder when applied to Pavia dataset

From Figure 6.6, we firstly notice that latent space dimension size 2 gives the best results for all the algorithms considered. We also observe that DEC has the best lower dimensional representational power as it has the highest Fisher's discriminant ratio values for all latent space dimension sizes.

Next, from Figures 6.6 and 6.7, we do not see a clear winner in terms of latent space dimension size, therefore we use the results from Figure 6.6 and conclude that indeed latent space dimension size of 2 gives the best results. We use this to report final results in this work.

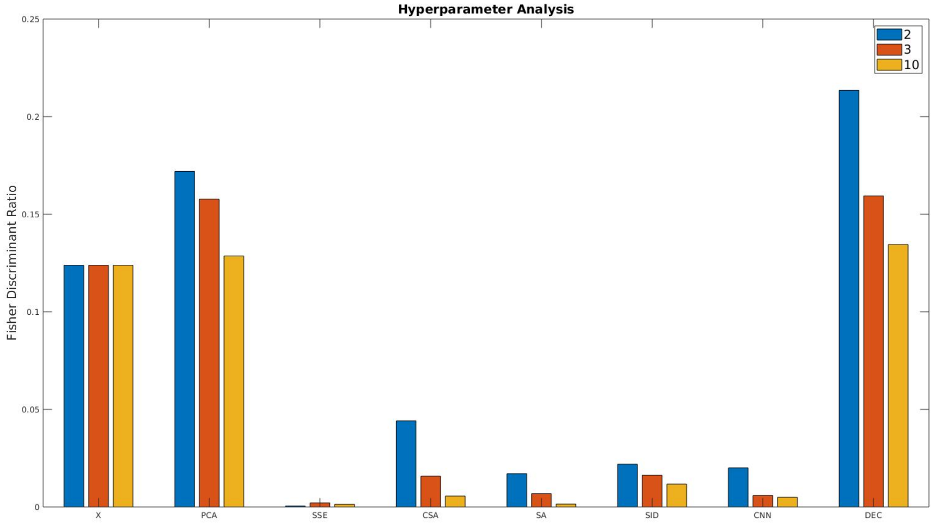


Figure 6.6: Hyperparameter analysis based on different number of latent space dimensions based on Fisher's discriminant ratio for various dimensionality reduction techniques when applied to Pavia dataset

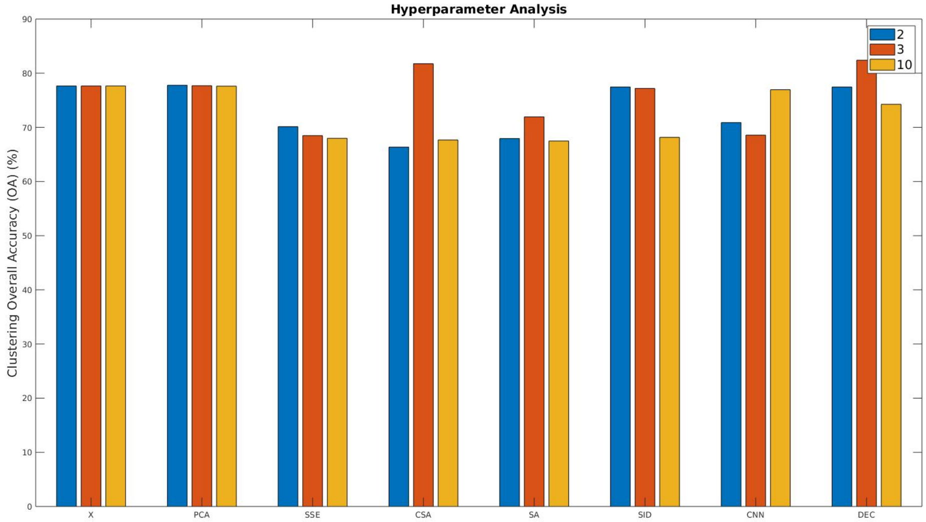


Figure 6.7: Hyperparameter analysis based on different number of latent space dimensions based on overall accuracy for various dimensionality reduction techniques when applied to Pavia dataset

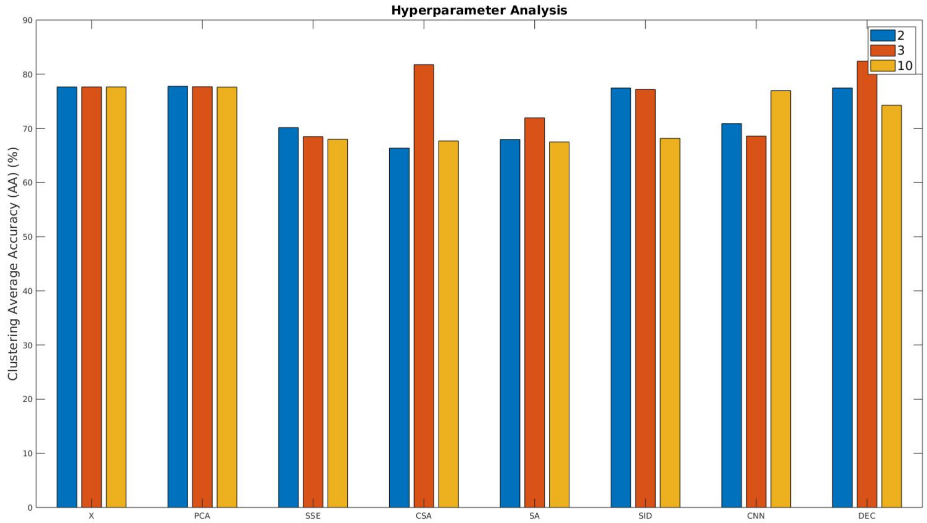


Figure 6.8: Hyperparameter analysis based on different number of latent space dimensions based on average accuracy for various dimensionality reduction techniques when applied to Pavia dataset

# Chapter 7

# Conclusion

## 7.1 Summary of Work

Data clustering helps to understand the inherent patterns and structures present in data. It is also a way to discover new patterns in unlabelled datasets and is crucial when we are working with complex unlabelled datasets.

Clustering techniques have certainly been a focus in the machine learning community and there have been several advancements in this area. Each clustering algorithm has its advantages and disadvantages. In this work, we comprehensively study clustering algorithms when applied to hyperspectral datasets. We delve into the theoretical and empirical differences between some of the most commonly used clustering algorithms.

Firstly, we begin with the definition of clustering and what hyperspectral images are. Then, we elaborate the challenges faced while clustering hyperspectral images. Further, we go on to look at the literature for clustering in the machine learning and remote sensing communities. The thesis then elaborates the process of various clustering techniques and glances over the salient features of these techniques. We have included a thorough theoretical analysis of the clustering techniques considered in this work.

Secondly, we conduct experiments over three commonly used hyperspectral datasets: Salinas Valley, Pavia and Indian Pines datasets. We scrutinize how each clustering algorithm performs on all of these datasets. We conclude that both spectral and spatial information are important for clustering hyperspectral images; and show that DLSS achieves the best performance over all datasets and is the most robust algorithm out of all.

Thirdly, we review dimensionality reduction techniques and look for techniques that have a higher representation power for clustering hyperspectral images. We analyse various techniques numerically using Fisher's discriminant ratio and also using representational tools like t-SNE. We observe that latent space dimension size of 2 has the highest representational power of all latent space dimension sizes considered.

Fourthly, we study the effect of change in hyperspectral dataset size in terms of number of clusters, spatial and spectral resolution on existing unsupervised machine learning techniques. We do so by using hyperspectral datasets of different spatial and spectral resolution and also containing different number of classes. We provide the overall accuracy, average accuracy and run time measurements for this analysis. It is observed that clustering performance deteriorates for higher number of clusters and class overlap.

Finally, we visit the impact of including or excluding unknown class data to unsupervised machine learning in the context of hyperspectral datasets. The hyperspectral datasets considered have a high class imbalance and the unknown class is the largest class in all of these datasets. For examples, for the Pavia dataset there are approximately 18000 sample points in the entire dataset, out of which only around 2000 are the known classes. It is difficult to find patterns in the unknown class as it is a mixture of several classes which calls for techniques that look for more number of classes than present in the ground truth. We also need better evaluation techniques to evaluate the final clustering result in the case where we look for more number of classes than present originally.

## 7.2 Future Work

There are several open challenges and questions to consider.

Firstly, there are very few hyperspectral datasets present for experimentation which limits the development of better clustering techniques. Looking at the currrent research in this area, we observe that the accuracies obtained by various techniques have somewhat saturated. There are improvements being made but their is scale is very limited. There is a need to learn from limited training data in this field. One way to mitigate this problem is by using transfer learning [82]. Other techniques that can be employed which have been overlooked are data augmentation and using multi-modal data where information from LiDAR, digital elevation models (DEMs), etc. are used.

Within, these datasets, there is a class imbalance problem where there is significantly more unknown class samples than known class samples. We propose that clustering techniques should naturally look for more classes than are present in the ground truth. This will help in discovering new patterns and also successfully employ the important information present in the unknown class.

Thirdly, deep learning techniques are overtaking over traditional clustering techniques but they lack interpretability and there is a need to better understand the black box that is a neural network in the field of remote sensing.

In this work, we looked at spectral learning which produced spatially inconsistent results. Algorithms like Spectral-Spatial Diffusion Learning (DLSS) and Sparse Manifold Clustering and Embedding (SMCE) that employ spatial information performed better than the rest on hyperspectral data. We also empirically looked at a 1-D convolutional autoencoder and would like to extend our experiments to 3-D convolutional autoencoders. We theorize that the use of spatial information in the form of 3-D patches would give better results than a 1-D convolutional autoencoder. A 3-D convolutional autoencoder would perform better because it captures the information present in neighboring pixels by forming patches. We know that similar pixels are present close to each other and therefore using 3-D patches will help reduce the salt and pepper noise as seen in autoencoders in Section 6.3 and in turn improve the clustering result.

We would also like to extend our work to hyperspectral super-resolution as seen in Section 3.4. Specifically, we would like to implement and compare our work with unsupervised CNN-based hyperspectral super-resolution as proposed by Fu et al.

Finally, this work has examined all the relevant methods of spectral analysis some of which incorporate spatial information, and concludes that the promise of spatial-spectral analysis, which might be achieved with CNNs or other methods, is important to pursue with larger, higher-resolution datasets that will become available in the future.

# Bibliography

[1] Introduction to autoencoders. URL https://www.jeremyjordan.me/autoencoders. (document), 4.2

[2] Aviris. URL https://aviris.jpl.nasa.gov. 2

[3] Landsat. URL https://www.usgs.gov/land-resources/nli/landsat. 2

[4] Surface reflectance calibration of terrestrial imaging spectroscopy data: a tutorial using aviris, 2020. URL https://archive.usgs.gov/archive/sites/speclab.cr.usgs.gov/PAPERS.calibration.tutorial/index.html. (document), 2.6, 2.2

[5] N. Acito, G. Corsini, and M. Diani. An unsupervised algorithm for hyperspectral image segmentation based on the gaussian mixture model. In *IGARSS 2003. 2003 IEEE International Geoscience and Remote Sensing Symposium. Proceedings (IEEE Cat. No.03CH37477)*, volume 6, pages 3745–3747 vol.6, July 2003. doi: 10.1109/IGARSS.2003.1295256. 3.1, 4.7

[6] N. Akhtar, F. Shafait, and A. Mian. Bayesian sparse representation for hyperspectral image super resolution. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3631–3640, 2015. 3.4

[7] Naveed Akhtar, Faisal Shafait, and Ajmal Mian. Sparse spatio-spectral representation for hyperspectral image super-resolution. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision – ECCV 2014*, pages 63–78, Cham, 2014. Springer International Publishing. ISBN 978-3-319-10584-0. 3.4

[8] Naveed Akhtar, Faisal Shafait, and Ajmal Mian. Hierarchical beta process with gaussian process prior for hyperspectral image super resolution. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision – ECCV 2016*, pages 103–120, Cham, 2016. Springer International Publishing. ISBN 978-3-319-46487-9. 3.4

[9] Haikel Alhichri, Nassim Ammour, Naif Alajlan, and Yakoub Bazi. Clustering of hyperspectral images with an ensemble method based on fuzzy c-means and markov random fields. *Arabian Journal for Science and Engineering*, 39:

3747–3757, 05 2014. doi: 10.1007/s13369-014-1037-3. 3.1

[10] Elie Aljalbout, Vladimir Golkov, Yawar Siddiqui, and Daniel Cremers. Clustering with deep learning: Taxonomy and new methods. *CoRR*, abs/1801.07648, 2018. URL http://arxiv.org/abs/1801.07648. 6.2.2

[11] Mihael Ankerst, Markus M. Breunig, Hans-Peter Kriegel, and Jörg Sander. Optics: Ordering points to identify the clustering structure. In *Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data*, SIGMOD 99, page 4960, New York, NY, USA, 1999. Association for Computing Machinery. ISBN 1581130848. doi: 10.1145/304182.304187. URL https://doi.org/10.1145/304182.304187. 3.1

[12] Marion F. Baumgardner, Larry L. Biehl, and David A. Landgrebe. 220 band aviris hyperspectral image data set: June 12, 1992 indian pine test site 3, Sep 2015. URL https://purr.purdue.edu/publications/1947/1. (document), 2.3.3, 2.10, 2.11

[13] Eyal Ben-Dor, K. Patkin, A. Banin, and Arnon Karnieli. Mapping of several soil properties using dais-7915 hyperspectral scanner data - a case study over soils in israel. *International Journal of Remote Sensing - INT J REMOTE SENS*, 23:1043–1062, 03 2002. doi: 10.1080/01431160010006962. 2.1

[14] James C. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms.* Kluwer Academic Publishers, USA, 1981. ISBN 0306406713. 3.1

[15] G. Camps-Valls, L. Gomez-Chova, J. Munoz-Mari, J. Vila-Frances, and J. Calpe-Maravilla. Composite kernels for hyperspectral image classification. *IEEE Geoscience and Remote Sensing Letters*, 3(1):93–97, 2006. 3.2

[16] W. R. Casper and Balu Nadiga. A new spectral clustering algorithm. *CoRR*, abs/1710.02756, 2017. URL http://arxiv.org/abs/1710.02756. 3.1

[17] David M. Chan, Roshan Rao, Forrest Huang, and John F. Canny. t-sne-cuda: Gpu-accelerated t-sne and its applications to modern data. *CoRR*, abs/1807.11824, 2018. URL http://arxiv.org/abs/1807.11824. 6.4

[18] Chein-I Chang. An information-theoretic approach to spectral variability, similarity, and discrimination for hyperspectral image analysis. *Information Theory, IEEE Transactions on*, 46:1927 – 1932, 09 2000. doi: 10.1109/18.857802. 3.2

[19] Chein-I Chang and Hsuan Ren. An experiment-based quantitative and comparative analysis of target detection and image classification algorithms for hyperspectral imagery. *IEEE Transactions on Geoscience and Remote Sensing*, 38(2):1044–1063, 2000. 3.2

[20] C. Chen, Y. Li, W. Liu, and J. Huang. Image fusion with local spectral

consistency and dynamic gradient sparsity. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2760–2765, 2014. 3.4

[21] Roger N. Clark. Imaging spectroscopy: Earth and planetary remote sensing with the USGS Tetracorder and expert systems. *Journal of Geophysical Research*, 108(E12):5131, 2003. ISSN 0148-0227. doi: 10.1029/2002JE001847. 1.1

[22] R. R. Coifman, S. Lafon, A. B. Lee, M. Maggioni, B. Nadler, F. Warner, and S. W. Zucker. Geometric diffusions as a tool for harmonic analysis and structure definition of data: Diffusion maps. *Proceedings of the National Academy of Sciences*, 102(21):7426–7431, 2005. ISSN 0027-8424. doi: 10.1073/pnas.0500334102. URL https://www.pnas.org/content/102/21/7426. 3.1, 4.10.1

[23] Ronald R. Coifman and Stphane Lafon. Diffusion maps. *Applied and Computational Harmonic Analysis*, 21(1):5 – 30, 2006. ISSN 1063-5203. doi: https://doi.org/10.1016/j.acha.2006.04.006. URL http://www.sciencedirect.com/science/article/pii/S1063520306000546. Special Issue: Diffusion Maps and Wavelets. 3.1

[24] Roberto Colombo, M. Merom, Andrea Marchesi, Lorenzo Busetto, Micol Rossini, Claudia Giardino, and C. Panigada. Estimation of leaf and canopy water content in poplar plantations by means of hyperspectral indices and inverse modeling. *Remote Sensing of Environment*, 112:1820–1834, 04 2008. doi: 10.1016/j.rse.2007.09.005. 2.1

[25] D. Comaniciu and P. Meer. Mean shift: a robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24 (5):603–619, May 2002. ISSN 1939-3539. doi: 10.1109/34.1000236. 3.1

[26] Surface Optics Corporation. Precision Agriculture and Hyperspectral Sensors: Monitoring Against Drought, Disease, and Nutrient Stress. URL https://surfaceoptics.com/applications/precision-agriculture-hyperspectral-sensors/. 2.1

[27] Rosa Correa Pabon and Carlos Souza Filho. Spectroscopic characterization of red latosols contaminated by petroleum-hydrocarbon and empirical model to estimate pollutant content and type. *Remote Sensing of Environment*, 175: 323–336, 03 2016. doi: 10.1016/j.rse.2016.01.005. 2.1

[28] Wojciech Czaja, Benjamin Manning, Lance McLean, and James M. Murphy. Fusion of aerial gamma-ray survey and remote sensing data for a deeper understanding of radionuclide fate after radiological incidents: examples from the fukushima dai-ichi response. *Journal of Radioanalytical and Nuclear Chemistry*, 307(3):2397–2401, Mar 2016. ISSN 1588-2780. doi: 10.1007/s10967-015-4650-z. URL https://doi.org/10.1007/s10967-015-4650-z. 4.10.1

[29] Mauro Dalla Mura, Jon Benediktsson, Bjorn Waske, and Lorenzo Bruzzone. Extended profiles with morphological attribute filters for the analysis of hyperspectral data. *International Journal of Remote Sensing - INT J REMOTE SENS*, 31:5975–5991, 12 2010. doi: 10.1080/01431161.2010.512425. 3.2

[30] R. N. Dave and K. Bhaswan. Adaptive fuzzy c-shells clustering and detection of ellipses. *IEEE Transactions on Neural Networks*, 3(5):643–662, 1992. 3.1

[31] Fernando De la Torre and Takeo Kanade. Discriminative cluster analysis. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, pages 241–248, New York, NY, USA, 2006. ACM. ISBN 1-59593-383-2. doi: 10.1145/1143844.1143875. URL http://doi.acm.org/10.1145/1143844.1143875. 4.4

[32] R. Dian, L. Fang, and S. Li. Hyperspectral image super-resolution via non-local sparse tensor factorization. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3862–3871, 2017. 3.4

[33] R. Dian, S. Li, A. Guo, and L. Fang. Deep hyperspectral image sharpening. *IEEE Transactions on Neural Networks and Learning Systems*, 29(11):5345–5355, 2018. 3.4

[34] David Donoho. High-dimensional data analysis: The curses and blessings of dimensionality. *AMS Math Challenges Lecture*, pages 1–32, 01 2000. 3.2

[35] Sarah Dunagan, Martha Gilmore, and Johan Varekamp. Effects of mercury on visible/near-infrared reflectance spectra of mustard spinach plants (brassica rapa p.). *Environmental pollution (Barking, Essex : 1987)*, 148:301–11, 08 2007. doi: 10.1016/j.envpol.2006.10.023. 2.1

[36] J. C. Dunn. A fuzzy relative of the isodata process and its use in detecting compact well-separated clusters. 1973. 3.1

[37] Ehsan Elhamifar and René Vidal. Sparse manifold clustering and embedding. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 55–63. Curran Associates, Inc., 2011. URL http://papers.nips.cc/paper/4246-sparse-manifold-clustering-and-embedding.pdf. 3.1, 4.5, 4.5

[38] Martin Ester, Hans-Peter Kriegel, Jrg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. pages 226–231. AAAI Press, 1996. 3.1, 4.9

[39] Hamid Ezzatabadi Pour and Saeid Homayouni. Clustering of hyperspectral image using fuzzy c-means based on spectral similarity measures. *Computations and Materials in Civil Engineering*, 1:47–54, 04 2016. 3.1

[40] J. Farifteh, F.D. Meer, Clement Atzberger, and Emmanuel John Carranza.

Quantitative analysis of salt-affected soil reflectance spectra: A comparison of two adaptive methods (plsr and ann). *Remote Sensing of Environment*, 110: 5978, 04 2007. doi: 10.1016/j.rse.2007.02.005. 2.1

[41] M. Fauvel, J. Chanussot, J. A. Benediktsson, and J. R. Sveinsson. Spectral and spatial classification of hyperspectral data using svms and morphological profiles. In *2007 IEEE International Geoscience and Remote Sensing Symposium*, pages 4834–4837, 2007. 3.2

[42] 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. 3.2

[43] Baowei Fei. Chapter 3.6 - hyperspectral imaging in medical applications. In Jos Manuel Amigo, editor, *Hyperspectral Imaging*, volume 32 of *Data Handling in Science and Technology*, pages 523 – 565. Elsevier, 2020. doi: https://doi.org/10.1016/B978-0-444-63977-6.00021-3. URL `http://www.sciencedirect.com/science/article/pii/B9780444639776000213`. 2.1

[44] Jonas Franke, Gunter Menz, Erich-Christian Oerke, and Uwe Rascher. Comparison of multi-and hyperspectral imaging data of leaf rust infected wheat plants. volume 5976, 09 2005. doi: 10.1117/12.626531. 2.1

[45] Y. Fu, T. Zhang, Y. Zheng, D. Zhang, and H. Huang. Hyperspectral image super-resolution with optimized rgb guidance. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11653–11662, 2019. 3.4, 7.2

[46] Alberto Garca-Gonzlez, Antonio Huerta, Sergio Zlotnik, and Pedro Dez. A kernel principal component analysis (kpca) digest with a new backward mapping (pre-image reconstruction) strategy, 2020. 3.1

[47] Alberto Candela Garza. Adaptive spectroscopic exploration driven by science hypotheses for geologic mapping. Master's thesis, Carnegie Mellon University, Pittsburgh, PA, August 2017. (document), 2.2

[48] Nicolas Gillis. The why and how of nonnegative matrix factorization, 2014. 4.8, 4.8

[49] Ccile Gomez, Raphael Viscarra Rossel, and Alex Mcbratney. Soil organic carbon prediction by hyperspectral remote sensing and field vis-nir spectroscopy: An australian case study. *Geoderma*, 146:403–411, 08 2008. doi: 10.1016/j.geoderma.2008.06.011. 2.1

[50] S. Guha, R. Rastogi, and K. Shim. Rock: a robust clustering algorithm for categorical attributes. In *Proceedings 15th International Conference on Data Engineering (Cat. No.99CB36337)*, pages 512–521, March 1999. doi: 10.1109/ICDE.1999.754967. 3.1

[51] Sudipto Guha, Rajeev Rastogi, and Kyuseok Shim. Cure: An efficient clustering algorithm for large databases. In *Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data*, SIGMOD 98, page 7384, New York, NY, USA, 1998. Association for Computing Machinery. ISBN 0897919955. doi: 10.1145/276304.276312. URL https://doi.org/10.1145/276304.276312. 3.1

[52] Xifeng Guo, Long Gao, Xinwang Liu, and Jianping Yin. Improved deep embedded clustering with local structure preservation. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pages 1753–1759, 2017. doi: 10.24963/ijcai.2017/243. URL https://doi.org/10.24963/ijcai.2017/243. 3.3

[53] L. Hamlin, R. O. Green, P. Mouroulis, M. Eastwood, D. Wilson, M. Dudik, and C. Paine. Imaging spectrometer science measurements for terrestrial ecology: AVIRIS and new developments. *IEEE Aerospace Conference Proceedings*, pages 1–7, 2011. 2

[54] L. Hamlin, R. O. Green, P. Mouroulis, M. Eastwood, D. Wilson, M. Dudik, and C. Paine. Imaging spectrometer science measurements for terrestrial ecology: Aviris and new developments. In *2011 Aerospace Conference*, pages 1–7, March 2011. doi: 10.1109/AERO.2011.5747395. 2

[55] Benedikt Hufnagl and Hans Lohninger. A graph-based clustering method with special focus on hyperspectral imaging. *Analytica Chimica Acta*, 1097:37 – 48, 2020. ISSN 0003-2670. doi: https://doi.org/10.1016/j.aca.2019.10.071. URL http://www.sciencedirect.com/science/article/pii/S000326701931311X. 3.1

[56] S. C. Jay, R. L. Lawrence, K. S. Repasky, and L. J. Rew. Detection of leafy spurge using hyper-spectral-spatial-temporal imagery. In *2010 IEEE International Geoscience and Remote Sensing Symposium*, pages 4374–4376, 2010. 2.1

[57] George Joseph and Jeganathan Chockalingam. *Fundamentals of Remote Sensing*. 11 2017. ISBN 978 93 86235 46 6. 2

[58] Martin Kanning, Insa Khling, Dieter Trautz, and Thomas Jarmer. High-resolution uav-based hyperspectral imagery for lai and chlorophyll estimations from wheat for yield prediction. *Remote Sensing*, 10:2000, 12 2018. doi: 10.3390/rs10122000. (document), 2.1, 2.4, 2.5

[59] Tapas Kanungo, David M. Mount, Nathan S. Netanyahu, Christine D. Piatko, Ruth Silverman, and Angela Y. Wu. An efficient k-means clustering algorithm: Analysis and implementation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24:881–892, 2002. 3.1, 4.1

[60] G. Karypis, Eui-Hong Han, and V. Kumar. Chameleon: hierarchical clustering using dynamic modeling. *Computer*, 32(8):68–75, Aug 1999. ISSN 1558-0814. doi: 10.1109/2.781637. 3.1

[61] R. Kawakami, Y. Matsushita, J. Wright, M. Ben-Ezra, Y. Tai, and K. Ikeuchi. High-resolution hyperspectral imaging via matrix factorization. In *CVPR 2011*, pages 2329–2336, 2011. 3.4

[62] S. Lafon, Y. Keller, and R. R. Coifman. Data fusion and multicue data matching by diffusion maps. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(11):1784–1797, Nov 2006. ISSN 1939-3539. doi: 10.1109/TPAMI.2006.223. 4.10.1

[63] C. Lanaras, E. Baltsavias, and K. Schindler. Hyperspectral super-resolution by coupled spectral unmixing. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 3586–3594, 2015. 3.4

[64] R. R. Lederman, R. Talmon, H. Wu, Y. Lo, and R. R. Coifman. Alternating diffusion for common manifold learning with application to sleep stage assessment. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5758–5762, April 2015. doi: 10.1109/ICASSP.2015.7179075. 4.10.1

[65] C. Lee and D. A. Landgrebe. Analyzing high-dimensional multispectral data. *IEEE Transactions on Geoscience and Remote Sensing*, 31(4):792–800, July 1993. ISSN 1558-0644. doi: 10.1109/36.239901. 3.2

[66] J. Li, P. R. Marpu, A. Plaza, J. M. Bioucas-Dias, and J. A. Benediktsson. Generalized composite kernel framework for hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 51(9):4816–4829, 2013. 3.2

[67] J. Li, X. Huang, P. Gamba, J. M. Bioucas-Dias, L. Zhang, J. A. Benediktsson, and A. Plaza. Multiple feature learning for hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 53(3):1592–1606, 2015. 3.2

[68] S. Li, R. Dian, L. Fang, and J. M. Bioucas-Dias. Fusing hyperspectral and multispectral images via coupled sparse tensor factorization. *IEEE Transactions on Image Processing*, 27(8):4118–4130, 2018. 3.4

[69] Wentian Li, Jane E Cerise, Yaning Yang, and Henry Han. Application of t-sne to human genetic data. *bioRxiv*, 2017. doi: 10.1101/114884. URL `https://www.biorxiv.org/content/early/2017/03/08/114884`. 6.4

[70] G. Licciardi, P. R. Marpu, J. Chanussot, and J. A. Benediktsson. Linear versus nonlinear pca for the classification of hyperspectral data based on the extended morphological profiles. *IEEE Geoscience and Remote Sensing Letters*, 9(3):

447–451, May 2012. ISSN 1558-0571. doi: 10.1109/LGRS.2011.2172185. 3.2

[71] L. Loncan, L. B. de Almeida, J. M. Bioucas-Dias, X. Briottet, J. Chanussot, N. Dobigeon, S. Fabre, W. Liao, G. A. Licciardi, M. Simes, J. Tourneret, M. A. Veganzones, G. Vivone, Q. Wei, and N. Yokoya. Hyperspectral pansharpening: A review. *IEEE Geoscience and Remote Sensing Magazine*, 3(3):27–46, 2015. 3.4

[72] Guolan Lu and Baowei Fei. Medical hyperspectral imaging: a review. *Journal of Biomedical Optics*, 19(1):1 – 24, 2014. doi: 10.1117/1.JBO.19.1.010901. URL https://doi.org/10.1117/1.JBO.19.1.010901. 2.1

[73] Mauro Maggioni and James M. Murphy. Learning by unsupervised nonlinear diffusion. *ArXiv*, abs/1810.06702, 2018. 1, 3

[74] Anne-Katrin Mahlein, Ulrike Steiner, Christian Hillnhtter, H.-W Dehne, and E.-C Oerke. Hyperspectral imaging for small-scale analysis of symptoms caused by different sugar beet disease. *Plant methods*, 8:3, 01 2012. doi: 10.1186/1746-4811-8-3. 2.1

[75] Zhaoyi Meng, Ekaterina Merkurjev, Alice Koniges, and Andrea L. Bertozzi. Hyperspectral Image Classification Using Graph Clustering Methods. *Image Processing On Line*, 7:218–245, 2017. doi: 10.5201/ipol.2017.204. 3.1

[76] L. Mou, P. Ghamisi, and X. X. Zhu. Unsupervised spectralspatial feature learning via deep residual convdeconv network for hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 56(1):391–406, Jan 2018. ISSN 1558-0644. doi: 10.1109/TGRS.2017.2748160. 3.2

[77] Nairouz Mrabah, Mohamed Bouguessa, and Riadh Ksantini. Adversarial deep embedded clustering: on a better trade-off between feature randomness and feature drift. *ArXiv*, abs/1909.11832, 2019. 3.3

[78] Nairouz Mrabah, Naimul Mefraz Khan, and Riadh Ksantini. Deep clustering with a dynamic autoencoder. *CoRR*, abs/1901.07752, 2019. URL http://arxiv.org/abs/1901.07752. 3.3

[79] Nairouz Mrabah, Naimul Mefraz Khan, and Riadh Ksantini. Deep clustering with a dynamic autoencoder. *CoRR*, abs/1901.07752, 2019. URL http://arxiv.org/abs/1901.07752. 5.2

[80] James M. Murphy and Mauro Maggioni. Unsupervised geometric learning of hyperspectral images. *CoRR*, abs/1704.07961, 2017. URL http://arxiv.org/abs/1704.07961. 2.3.2, 2.4, 3.1, 4.10, 5.2, 5.3, 6.2.1

[81] Jakub Nalepa, Michal Myller, Yasuteru Imai, Ken-ichi Honda, Tomomi Takeda, and Marek Antoniak. Unsupervised segmentation of hyperspectral images using 3d convolutional autoencoders. *CoRR*, abs/1907.08870, 2019. URL

http://arxiv.org/abs/1907.08870. 3.2

[82] Jakub Nalepa, Michal Myller, and Michal Kawulok. Transfer learning for segmenting dimensionally-reduced hyperspectral images. *CoRR*, abs/1906.09631, 2019. URL http://arxiv.org/abs/1906.09631. 7.2

[83] Andrew Y Ng, Michael I Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in neural information processing systems*, pages 849–856, 2002. 4.5

[84] Roope Nsi, Eija Honkavaara, Minna Blomqvist, Lyytikainen-Saarenmaa Paivi, Teemu Hakala, Niko Viljanen, Kantola Tuula, and Markus Holopainen. Remote sensing of bark beetle damage in urban forests at individual tree level using a novel hyperspectral camera from uav and aircraft. *Urban Forestry Urban Greening*, 30, 01 2018. doi: 10.1016/j.ufug.2018.01.010. 2.1

[85] S.L. Osborne, J.s Schepers, D. Francis, and M.R. Schlemmer. Detection of phosphorus and nitrogen deficiencies in corn using spectral radiance measurements. *Agronomy Journal*, 94(6), 11 2002. doi: 10.2134/agronj2002.1215. 2.1

[86] Hae-Sang Park and Chi-Hyuck Jun. A simple and fast algorithm for k-medoids clustering. *Expert Syst. Appl.*, 36:3336–3341, 03 2009. doi: 10.1016/j.eswa.2008.01.039. 3.1

[87] Hongbin Pu, Dan Liu, Jia-Huan Qu, and Da-Wen Sun. Applications of imaging spectrometry in inland water quality monitoringa review of recent developments. *Water, Air, Soil Pollution*, 228, 04 2017. doi: 10.1007/s11270-017-3294-8. 2.1

[88] Ying Qu, Hairong Qi, and Chiman Kwan. Unsupervised sparse dirichlet-net for hyperspectral image super-resolution, 2018. 3.4

[89] Craig Rodarmel and Jie Shan. Principal component analysis for hyperspectral image classification. 2002. 3.2

[90] Alex Rodriguez and Alessandro Laio. Clustering by fast search and find of density peaks. *Science*, 344(6191):1492–1496, 2014. ISSN 0036-8075. doi: 10.1126/science.1242072. URL http://science.sciencemag.org/content/344/6191/1492. 4.9

[91] Guglielmo Rossi, Federico Landini, Teresa Salvatici, Marco Romoli, Maurizio Pancrazzi, Mauro Focardi, Vladimiro Noce, Sandro Moretti, Nicola Casagli, and Cristian Baccani. Optical design of a hyperspectral drone advanced camera for soil monitoring using an electro-optical liquid crystal technology. page 20, 06 2018. doi: 10.1117/12.2311680. 2.1

[92] Sam T. Roweis and Lawrence K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000. ISSN 0036-8075. doi: 10.1126/science.290.5500.2323. URL https://science.sciencemag.org/

content/290/5500/2323. 3.1, 4.10.1

[93] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Parallel distributed processing: Explorations in the microstructure of cognition, vol. 1. chapter Learning Internal Representations by Error Propagation, pages 318–362. MIT Press, Cambridge, MA, USA, 1986. ISBN 0-262-68053-X. URL http://dl.acm.org/citation.cfm?id=104279.104293. 3.2

[94] Rebecca Scafutto, Carlos Souza Filho, and Benoit Rivard. Characterization of mineral substrates impregnated with crude oils using proximal infrared hyperspectral imaging. *Remote Sensing of Environment*, 179:116–130, 06 2016. doi: 10.1016/j.rse.2016.03.033. 2.1

[95] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *CoRR*, abs/1404.7828, 2014. URL http://arxiv.org/abs/1404.7828. 4.3

[96] Shao-Shan Chiang, Chein-I Chang, and I. W. Ginsberg. Unsupervised hyperspectral image analysis using independent component analysis. In *IGARSS 2000. IEEE 2000 International Geoscience and Remote Sensing Symposium. Taking the Pulse of the Planet: The Role of Remote Sensing in Managing the Environment. Proceedings (Cat. No.00CH37120)*, volume 7, pages 3136–3138 vol.7, July 2000. doi: 10.1109/IGARSS.2000.860361. 3.2

[97] Peg Shippert. Introduction to hyperspectral image analysis. *Online Journal of Space Communication*, 01 2003. (document), 2.3

[98] Jonathon Shlens. A tutorial on principal component analysis. *CoRR*, abs/1404.1100, 2014. URL http://arxiv.org/abs/1404.1100. 4.2

[99] C. Tao, J. Jin, Y. Tang, and Z. Zou. Hyperspectral imagery classification based on rotation invariant spectral-spatial feature. In *2013 IEEE International Geoscience and Remote Sensing Symposium - IGARSS*, pages 422–424, 2013. 3.2

[100] C. Tao, H. Pan, Y. Li, and Z. Zou. Unsupervised spectralspatial feature learning with stacked sparse autoencoder for hyperspectral imagery classification. *IEEE Geoscience and Remote Sensing Letters*, 12(12):2438–2442, Dec 2015. ISSN 1558-0571. doi: 10.1109/LGRS.2015.2482520. 3.2

[101] Chao Tao, Hongbo Pan, Yansheng Li, and Zhengrou Zou. Unsupervised spectral-spatial feature learning with stacked sparse autoencoder for hyprspectral imagery classification. *IEEE Geoscience and Remote Sensing Letters*, 12, 09 2015. doi: 10.1109/LGRS.2015.2482520. 3.2

[102] TNTmips. Introduction to hyperspectral imaging. URL https://www.microimages.com/documentation/Tutorials/hyprspec.pdf. (document), 2.2, 2.7

[103] Devis Tuia, Rmi Flamary, and Nicolas Courty. Multiclass feature learning for hyperspectral image classification: Sparse and hierarchical solutions. *ISPRS Journal of Photogrammetry and Remote Sensing*, 105:272 – 285, 2015. ISSN 0924-2716. doi: https://doi.org/10.1016/j.isprsjprs.2015.01.006. URL http://www.sciencedirect.com/science/article/pii/S0924271615000234. 3.2

[104] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 2008. URL http://www.jmlr.org/papers/v9/vandermaaten08a.html. 6.4

[105] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *J. Mach. Learn. Res.*, 11:3371–3408, December 2010. ISSN 1532-4435. URL http://dl.acm.org/citation.cfm?id=1756006.1953039. 4.6

[106] Iosif Vorovencii. The hyperspectral sensors used in satellite and aerial remote sensing. *Bulletin of the Transilvania University of Braov*, 2:51–56, 01 2009. 2

[107] Lloyd Windrim, Arman Melkumyan, Richard Murphy, Anna Chlingaryan, and Juan Nieto. Unsupervised feature learning for illumination robustness. 09 2016. doi: 10.1109/ICIP.2016.7533202. 3.2

[108] Lloyd Windrim, Rishi Ramakrishnan, Arman Melkumyan, Richard J Murphy, and Anna Chlingaryan. Unsupervised feature-learning for hyperspectral data with autoencoders. *Remote Sensing*, 11(7):864, 2019. 3.2, 4.3.3, 4.3.3, 6.2.1, 6.5.2

[109] Junyuan Xie, Ross B. Girshick, and Ali Farhadi. Unsupervised deep embedding for clustering analysis. *CoRR*, abs/1511.06335, 2015. URL http://arxiv.org/abs/1511.06335. 3.3, 4.6, 6.2.1, 6.5.2

[110] Xin Geng, De-Chuan Zhan, and Zhi-Hua Zhou. Supervised nonlinear dimensionality reduction for visualization and classification. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 35(6):1098–1107, 2005. 3.1

[111] R. R. Yager and D. P. Filev. Approximate clustering via the mountain method. *IEEE Transactions on Systems, Man, and Cybernetics*, 24(8):1279–1284, 1994. 3.1

[112] Jieping Ye, Zheng Zhao, and Mingrui Wu. Discriminative k-means for clustering. In J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 1649–1656. Curran Associates, Inc., 2008. URL http://papers.nips.cc/paper/3176-discriminative-k-means-for-clustering.pdf. 4.4

[113] Yizong Cheng. Mean shift, mode seeking, and clustering. *IEEE Transactions*

*on Pattern Analysis and Machine Intelligence*, 17(8):790–799, Aug 1995. ISSN 1939-3539. doi: 10.1109/34.400568. 4.9

[114] Roberta H. Yuhas, Alexander F. H. Goetz, and Joseph W. Boardman. Discrimination among semi-arid landscape endmembers using the spectral angle mapper (sam) algorithm. 1992. 3.2

[115] Tian Zhang, Raghu Ramakrishnan, and Miron Livny. Birch: An efficient data clustering method for very large databases, 1996. 3.1

[116] W. Zhu, V. Chayes, A. Tiard, S. Sanchez, D. Dahlberg, A. L. Bertozzi, S. Osher, D. Zosso, and D. Kuang. Unsupervised classification in hyperspectral imagery with nonlocal total variation and primal-dual hybrid gradient algorithm. *IEEE Transactions on Geoscience and Remote Sensing*, 55(5):2786–2798, May 2017. ISSN 0196-2892. doi: 10.1109/TGRS.2017.2654486. 6.2.1

[117] W. Zhu, V. Chayes, A. Tiard, S. Sanchez, D. Dahlberg, A. L. Bertozzi, S. Osher, D. Zosso, and D. Kuang. Unsupervised classification in hyperspectral imagery with nonlocal total variation and primal-dual hybrid gradient algorithm. *IEEE Transactions on Geoscience and Remote Sensing*, 55(5):2786–2798, May 2017. ISSN 1558-0644. doi: 10.1109/TGRS.2017.2654486. 6.2.2