Satellite Image Classification Using Deep Learning Techniques

A PROJECT REPORT

Submitted by

SARASIJ BASU MALLICK

[ROLL NO: 30098121003]

[REG. NO: 213001898110003 of 2021-2023]

Supervised by

Ms. Dipanwita Ghosh

Assistant Professor

Department of Information Technology

in partial fulfillment for the award of the degree

of

MASTER OF SCIENCE

IN

IT(DATA SCIENCE)

Year: 2023

Maulana Abul Kalam Azad University of Technology

NH-12 (Old NH-34) Simhat

Haringhata, Nadia 741249, West Bengal

BONAFIDE CERTIFICATE

Certified that this project report **"Satellite Image Classification Using Deep Learning Techniques"** is the bona fide work of "**Sarasij Basu Mallick"** who carried out the project work under my supervision.

SIGNATURE
Dr. Debasis Giri

HEAD OF THE DEPARTMENT
Department of Information Technology

SIGNATURE
Ms. Dipanwita Ghosh

ASSISTANT PROFESSOR
Department of Information Technology

SIGNATURE
External Examiner:

ACKNOWLEDGEMENTS

It gives me immense pleasure to express my deepest sense of gratitude and sincere thanks to my respected guide Ms. Dipanwita Ghosh (Assistant Professor), Department of Information Technology, MAKAUT, WEST BENGAL, for their valuable guidance, encouragement and help for completing this work. Their useful suggestions for this whole work and cooperative behaviour are sincerely acknowledged.

I also wish to express my indebtedness to my parents as well as my family members whose blessings and support always helped me to face the challenges ahead.

SARASIJ BASU MALLICK

Maulana Abul Kalam Azad University Of Technology,West Bengal

May,2023.

TABLE OF CONTENTS

Chapter No.	Title	Page no.
	List Of Figures	vi
	List Of Tables	vii
	Abstract	1
1	Introduction	2
2	Literature Review	5
3	Datasets	7
3.1	Indian Pines Dataset	7
3.2	Salinas Dataset	8
3.3	Samson Dataset	8
3.4	Jasper Ridge Dataset	9
3.5	Henry Island Dataset	10
4	Deep Learning	11
4.1	Perceptron	11
4.2	Multi-Layer Perceptron	12
4.3	Deep Neural Network	13
4.4	Convolutional Network – CNN	14
4.4.1	Dropout Layers	15
4.5	3D Convolutional Neural Network	15
4.5.1	Comparison between 3D Convolutional Neural Network and 2D Convolutional Neural Network	16
4.5.2	Hyperspectral Image Classification using 3D Convolutional Neural Network	18
5	Auto Encoders	20
5.1	Different Types of Auto Encoders	21
5.1.1	Under-Complete Auto Encoder	21
5.1.2	Over-Complete Auto Encoder	22
5.1.3	Few generic Auto Encoders	22
5.2	Classification of Satellite Image Using Auto Encoders	22
6	Proposed Methodology	24
6.1	Dense Neural Network	24
6.2	Auto Encoder + k nearest neighbor Architecture	24

6.3	3D Convolutional Neural Network Architecture	25
6.4	Accuracy Measurement Technique	26
7	Experimental Results and Analysis	27
7.1	Classified Output of Indian Pines Dataset in different models	27
7.2	Classified Output of Salinas Dataset in different models	28
7.3	Classified Output of Samson Dataset in different models	28
7.4	Classified Output of Jasper Ridge Dataset in different models	29
7.5	Classified Output of Henry Island in different models	29
7.6	Comparison in Validation Accuracy of the Models	30
8	Conclusion and Future Work	31
9	References	33

LIST OF FIGURES

FIGURE NO.	FIGURE NAME	PAGE NO.
3.1	Indian Pines Dataset	7
3.2	Salinas Dataset Image	8
3.3	Samson Dataset	9
3.4	Jasper Ridge Dataset	10
3.5	Henry Island Dataset	10
4.1	Perceptron	11
4.2	Perceptron Learning	12
4.3	Multi-Layer Perceptron	13
4.4	Deep Neural Network	14
5.1	Auto Encoder	20
6.1	Dense Neural Network Architecture	24
6.2	Auto Encoder Architecture	24
6.3	3D CNN Architecture	25
7.1	Classified Output of Indian Pines Dataset	27
7.2	Classified Output of Salinas Dataset	28
7.3	Classified Output of Samson Dataset	29
7.4	Classified Output of Jasper Ridge Dataset	29
7.5	Classified Output of Henry Island Dataset	30

LIST OF TABLES

FIGURE NO.	TABLE NAME	PAGE NO.
7.1	Comparison of Several Deep Learning Models	30

Abstract:

Multimedia applications and processing are essential in many artificial intelligence applications, including image classification, video summarization, and image retrieval. Convolutional neural networks and deep neural nets are commonly used in multimedia approaches to classify data without human intervention. The demand for mapping and GIS data for environmental assessment and monitoring is increasing, and a recent study proposes a method for satellite image classification using deep learning and feature extraction with convolutional neural networks.

The study used a standard dataset and modelled three classification methods: a simple Dense Neural Network [30], an Auto Encoder [28] + KNN [29] Classifier, and a 3D CNN [15]. The models were applied to classify portions of the region of interest, and the results were analysed and compared to existing classification methods. The findings suggest that this approach can provide an accurate and efficient classification of satellite images, with potential implications for environmental monitoring and assessment. Overall, the article presents a promising method for Satellite Image Classification using deep learning and Convolutional NN (neural net).

Keywords: Deep Learning, CNN,3d CNN, Satellite Image Classification.

Chapter 1

Introduction:

Within recent times, the field of remote sensing technology has experienced significant advancements, leading to increased accessibility to a vast array of high-resolution remote sensing imagery. As a result, numerous researchers in remote sensing detection and classification have transitioned from conventional methodologies to more modern techniques. This shift is driven by the increased availability of high-quality remote sensing data, enabling researchers to explore innovative approaches and achieve enhanced results in their detection and classification tasks.

The field of remote sensing technology has witnessed rapid development in recent years, revolutionizing the acquisition of extensive collections of high-resolution remote sensing imagery. This accessibility has prompted many researchers in remote sensing detection and classification to transition from traditional methods to newer, more advanced techniques. This paradigm shift is a direct response to the improved availability of high-quality remote sensing data, empowering researchers to explore innovative approaches and achieve superior outcomes in their detection and classification endeavours.

Image classification can be categorized into three primary classes based on their characteristics. The first class is the "handcrafted feature-based method," which focuses on utilizing attributes like colour and shape information that are inherent to the sensory image. The second class is the "unsupervised feature learning-based methods," which aims to learn a set of fundamental functions such as feature encoding. A common encoding method used is quantization, although a more effective approach is Fisher encoding. In Fisher encoding, the input consists of a set of handcrafted characteristics, and the output is a set of learned features. Finally, there is the "deep feature learning-based methods" class, commonly referred to as Deep Learning (DL)[27]. This class leverages deep neural networks to automatically learn and extract features from images.

DL has gained significant popularity due to its ability to achieve state-of-the-art performance in various image classification tasks.

Sample is used to encode features. There are many deep learning architectures, one of which is a CNN (Conv Neural Net). CNNs are widely used and have been used in recent years to solve problems as diverse and complex as image detection, recognition and classification using a sequence of transition layers.

Convolutional Neural Networks (CNNs) share similarities with traditional neural networks as they consist of interconnected neurons with learned weights and biases. These neurons receive inputs and perform non-linear computations, similar to a feedforward artificial neural network. However, the distinguishing feature of CNNs lies in their architecture, specifically designed for processing images.

CNNs leverage the concept of convolutions, which allow them to encode specific properties into their architecture. By applying convolutions to image inputs, CNNs can effectively capture spatial dependencies and local patterns within the images. This architectural design makes CNNs highly suitable for tasks such as image classification, where extracting meaningful features from images is crucial for accurate predictions.

A CNN typically follows a structured architecture comprising several layers, including the convolutional layer, pooling layer, and fully connected layers. It can be considered as a specialized variant of a neural network designed specifically for image processing tasks.

The convolutional layer plays a crucial role in a CNN by extracting low-level features from the input images. These features can include lines, edges, angles, and corners, enabling the network to capture essential visual patterns. The subsequent pooling layers help reduce the impact of noise and distortions in the images, enhancing the network's robustness.

Non-linear layers, acting as activation functions, contribute to distinguishing different features at each hidden layer. They introduce non-linear transformations to the learned features, allowing the network to model complex relationships within the data.

Finally, the fully connected layers aggregate the weighted information from the preceding feature layers. These layers summarize the extracted features mathematically, facilitating the network's ability to make accurate predictions or classifications based on the learned representations.

In summary, a CNN's structure includes convolutional layers for feature extraction, pooling layers for noise reduction, non-linear layers for activation, and fully connected layers for summarizing and decision-making based on the learned features.

In this topic we shall classify objects like river, building, vegetation, etc. We did classification task of some of the standard datasets. Like Indian Pines [22], Salinas Dataset [23], Samson Dataset [24], Jasper Ridge Dataset [25] and Henry Island dataset [26]. Then we compared the accuracy of these.

Chapter 2

Literature Review:

In the study [17] authors propose a split architecture that uses a pseudo-ground truth for abundances to guide the optimization of the "unmixing network" (UN). Prior to the UN, an "approximation network" (AN) is introduced to improve the association between the centre pixel and its neighbours, emphasizing spatial correlation in the abundances. The proposed Guided Encoder-Decoder Architecture for Hyperspectral Unmixing with Spatial Smoothness (GAUSS) uses one-hot encoded abundances as the pseudo-ground truth to guide the UN, which is computed using the k-means algorithm, eliminating the need for prior HU methods. Additionally, the study releases the single-layer constraint on the mixing network (MN) by introducing the UN-generated abundances instead of using the standard AE for HU [32] .The study experimented with two modifications to the pre-trained network using the GAUSS method. In GAUSSblind, the UN and MN were concatenated to back-propagate the reconstruction error gradients to the encoder. In GAUSSprime, the abundance results of a signal processing (SP) method with reliable abundance results were used as the pseudo-ground truth with the GAUSS architecture. The performance of the proposed architectures was compared to existing HU algorithms from both DL and SP domains for four experimental datasets. The results show that the proposed architectures either outperformed or equalled the performance of existing HU algorithms. Authors in [11] for efficient processing of large satellite images, focusing on improving processing efficiency, identifying data correlations, and extracting continuous features. The aim is to detect and locate rivers, roads, and major highways in satellite imagery obtained from remote earth-sensory observatories. The proposed approach involves real-time analysis of the ENVISAT satellite mission dataset. Statistical measures, RepTree machine learning classifiers, and Euclidean distance are employed to develop algorithms for continuous feature extraction and detection. To enhance system efficiency, the designed system utilizes the Hadoop ecosystem. It

encompasses several steps, including data collection, filtering, load balancing, processing, merging, and interpretation. To handle the large volume of satellite imagery, the system is implemented on an Apache Hadoop framework using the MapReduce programming model. This enables effective processing of the ASAR/ENVISAT mission datasets. Overall, the proposed system offers a real-time and efficient solution for processing and analyzing large satellite images, specifically for identifying rivers, roads, and major highways, leveraging statistical measures, machine learning classifiers, and the capabilities of the Hadoop ecosystem. In the study [10] , the authors explore the application of remote sensing and hyperspectral data for monitoring changes in mangrove forests and analyzing inter-species competition. Their objective is to investigate the interaction among different mangrove species within the Sunderban Delta region. The study area is characterized by natural disasters, human activities, and population growth, which have altered the availability of resources and the physicochemical environment, resulting in a competition for survival among the mangrove species. The authors utilize detailed information obtained from hyperspectral image data to analyze and understand the dynamics of the mangrove community. The paper also references previous studies that have focused on modeling the dynamic mangrove ecosystem by conducting physical monitoring of dense forests and ground truthing over an extended period of time.

Chapter 3

Datasets:

3.1 Indian Pines Dataset

The Indian Pines dataset is a commonly used hyperspectral remote sensing dataset that captures high-resolution imagery of agricultural fields in Indiana, USA. It was accumulated using the Airborne Visible/IR Imaging Spectrometer (AVIRIS)[31] sensor, which measures reflectance of light across various spectral bands. With 145 spectral bands covering a 145x145 pixel area, each pixel in the Indian Pines dataset represents a specific land cover class, such as corn, soybeans, wheat, or bare soil. Due to its comprehensive spectral information varying from the visible to the near-IR spectrum, this dataset is widely employed in research and applications related to hyperspectral image analysis. The Indian Pines dataset finds applications in land cover classification, crop monitoring, and vegetation mapping. Its extensive spectral data allows for detailed analysis of different land cover types and their spatial patterns, making it a valuable resource for studies in remote sensing, agriculture, and environmental monitoring.

Fig 3.1 : Indian Pines Dataset

3.2 Salinas Dataset

The Salinas dataset was acquired using an AVIRIS sensor with a higher spatial resolution of 3.7-meter pixels. This dataset covers the Valley of Salinas in California in USA. This dataset comprises imagery that represents sixteen distinct land cover classes, including various types of vegetables, vineyard fields, and bare soils, among others. This hyperspectral image provides detailed information across numerous spectral bands, allowing for the analysis and classification of different land cover types. Researchers and practitioners utilize the Salinas dataset for tasks for example land cover mapping, crop classification, agricultural monitoring, and studying the spatial distribution of various land features in the Salinas Valley. Its high spatial resolution and diverse class composition make it a valuable resource for remote sensing studies and applications related to agricultural and environmental analysis.

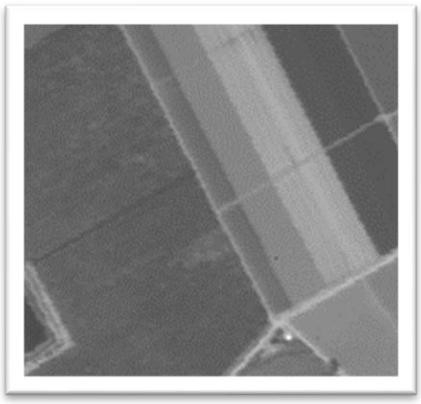


Fig 3.2 : Salinas Dataset Image

3.3 Samson Dataset

The Samson hyperspectral dataset is a widely used remote sensing dataset that captures hyperspectral imagery of an urban area in the city of Fort Collins, Colorado, USA. It was collected using the Reflective Optics System Imaging Spectrometer (ROSIS) sensor. The dataset consists of 156 bands (Spectral), covering a spatial extent of 95x95 pixels. Each pixel in the Samson dataset represents a specific material or land cover class, such as asphalt, meadow, bare soil, or trees. The spectral bands span the visible

to near-infrared spectrum, providing detailed information about the reflectance properties of different materials and land cover types. The Samson hyperspectral dataset is commonly used in research and applications related to hyperspectral image analysis, including land cover classification, urban monitoring, and material identification. Its relatively small size and urban focus make it suitable for studying the spectral signatures and distinguishing characteristics of various urban materials and land cover types. It serves as a valuable resource for understanding urban environments, analysing land use patterns, and developing remote sensing algorithms and techniques specific to urban areas.

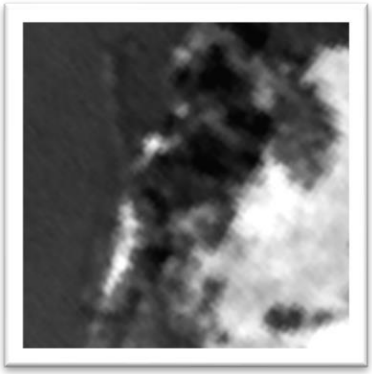


Fig 3.3: Samson Dataset

3.4 Jasper Ridge Dataset

The Jasper Ridge dataset is a well-known hyperspectral dataset captured using the AVIRIS instrument over a rural area in Jasper Ridge, California, USA. The dataset consists of a 100x100 pixel grid with 298 spectral bands, covering a wide wavelength range from 380nm to 2500nm. This extensive spectral coverage enables detailed analysis of reflectance properties across different parts of the electromagnetic spectrum.

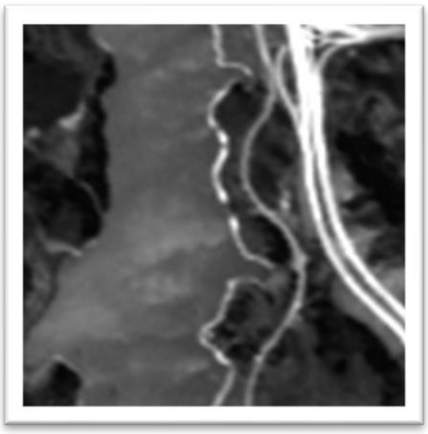


Fig 3.4: Jasper Ridge Dataset

3.5 Henry Island Dataset

EO-1 Hyperion hyperspectral data was acquired on 27th May 2011. This data represents different classes of mangroves like *Avicennia marina, Avicennia officinalis, Avicennia alba, Excoecaria agallocha, Ceriops decandra, Brugeria cylindrica, Phoenix paludosa etc.* In this study different deep learning-based classification methods have been used to classify these different mangrove communities.

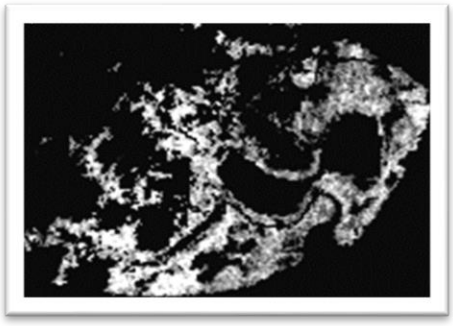


Fig 3.5 Henry Island Dataset

Chapter 4

Deep Learning:

For a concise introduction to deep learning, a recommended resource is the book authored by Goodfellow et al [9]. This comprehensive guide delves into the architecture of deep learning models, offering valuable insights into the selection process for specific architectures. By referring to this book, readers can gain a thorough understanding of the fundamental principles of deep learning and the rationale behind the choice of different model architectures.

4.1 Perceptron

The Perceptron, initially introduced by Minsky and Papert in 1969[19], is considered the fundamental model of neural networks. It consists of a single neuron that accepts inputs X_1, X_2, ..., X_n , each associated with a label of either 0 or 1. The output y is determined by the weighted sum of inputs. The primary objective of the perceptron model is to iteratively learn and optimize the weights (parameters) of the network to achieve accurate classification.

It is important to note that the perceptron model assumes linear separability of the inputs. The learning process involves adjusting the weights based on misclassified instances until convergence is reached. This iterative learning allows the perceptron to improve its ability to classify inputs correctly.

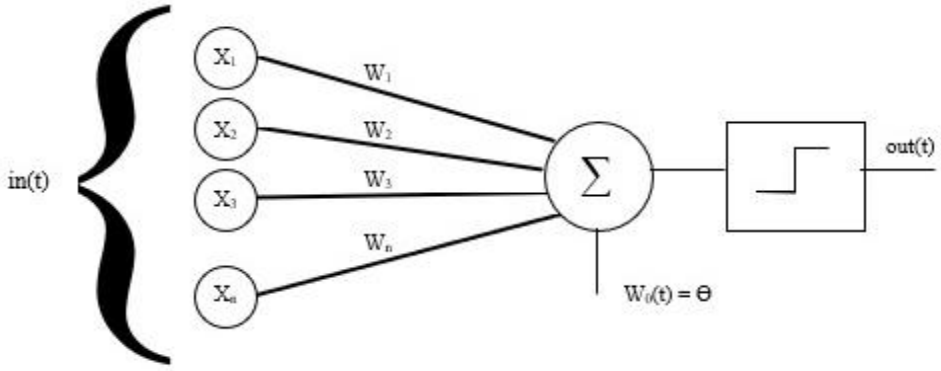


Fig 4.1: Perceptron

The perceptron learns weights via the algorithm mentioned as follows:

Algorithm: Perceptron Learning Algorithm

```
P ← inputs with label 1;
N ← inputs with label 0;
Initialize w randomly;
while !convergence do
        Pick random x ∈ P ∪ N ;
        if x∈ P and w.x < 0 then
        w = w+x
        end
        if x∈N and w .x ≥ 0
        w = w-x
        end
end
/ / the algorithm converges when all the
inputs are classified correctly
```

Fig 4.2: Perceptron Learning

4.2 Multi-Layer Perceptron

The limitations of the perceptron model became evident when it failed to learn even simple functions like XOR, leading to the development of more advanced neural network architectures, such as the multilayer perceptron (MLP) [13]. The MLP introduced the concept of the backpropagation algorithm, which was discovered by Geoffrey Hinton in 1986.

In the MLP, inputs are passed through multiple layers of neurons to generate an output. The network employs a loss function to calculate the error between the predicted output and the actual output. The error is then propagated back through the network using the

backpropagation algorithm. This involves adjusting the weights of the neurons based on the gradient descent update rule, which seeks to minimize the error.

The training process of an MLP is typically divided into epochs. In each epoch, the input data is iteratively fed forward through the network, and the resulting error is backpropagated to update the weights. The training continues until the error falls within a predefined tolerance level, a predetermined number of epochs have been completed, or the network reaches a point where it stops learning effectively.

This combination of forward propagation, error calculation, backpropagation, and weight adjustment form the basis of the MLP's learning process. It allows the network to iteratively refine its weights and learn complex relationships within the data, leading to improved accuracy in classification tasks.

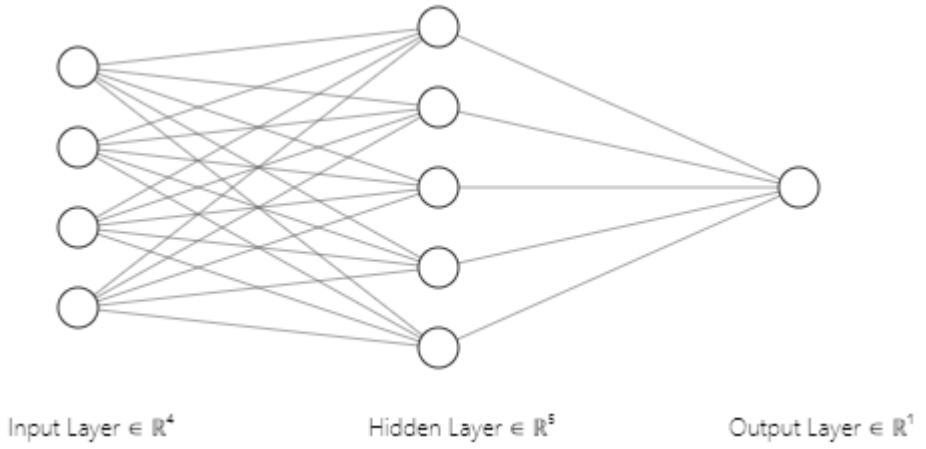


Fig 4.3: Multi-Layer Perceptron

4.3 Deep Neural Network

The discovery of the Universal Approximation Theorem demonstrated that any given function could be approximated by a neural network with a sufficient number of neurons. This finding motivated researchers to explore more complex network architectures. In multilayer perceptron, the layers between the input and output layers

are referred to as hidden layers. If the number of hidden layers is substantial, the network is referred to as a deep neural network. One major shortcoming was the significant computational resources required to train deep neural networks.

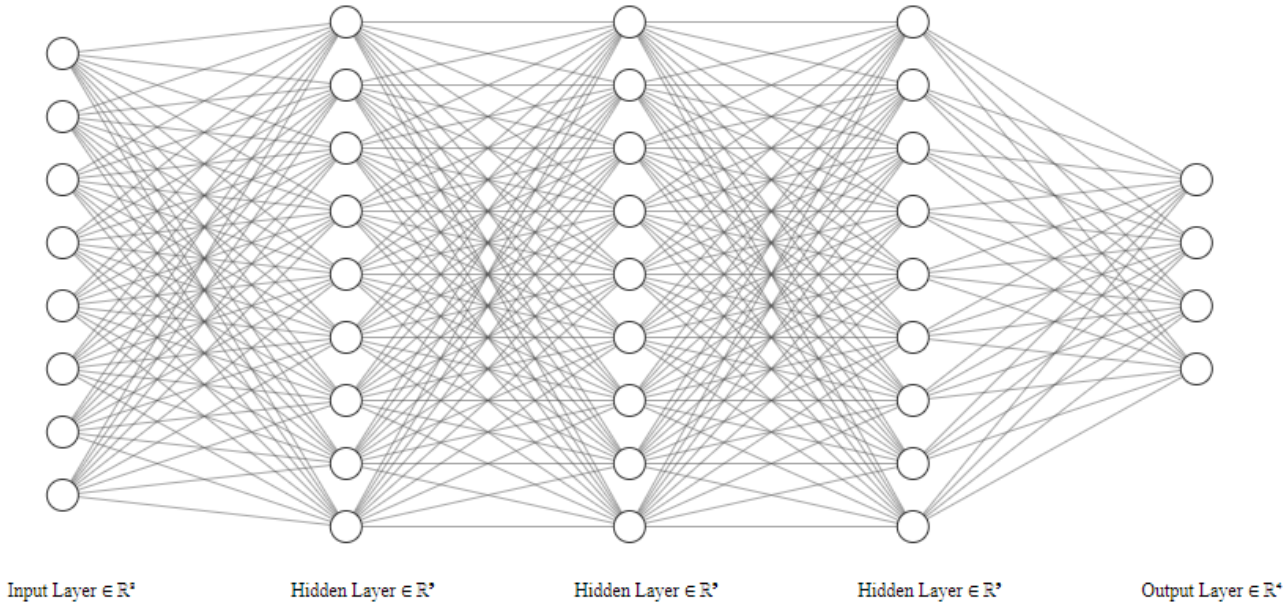


Fig 4.4: Deep Neural Network

4.4 Convolutional Neural Network - CNN

The invention of Convolutional Neural Network (CNN) [15] was an attempt to address problem of limited computational resources required to train deep neural networks. The approach involves using more filters along with convolving them along with the i/p data to learn representations that capture underlying principles. Convolutional neural networks offer two primary advantages:

- Parameter Sharing: The filter is utilized across whole parts of the input data. For example, filter that detects the vertical edges, is used throughout the image to identify vertical edges.
- Sparsity of Connections: The concept of sparsity of connections is characterized by the selective connectivity between neurons in each layer of a neural network. Unlike deep neural networks, where every neuron is

connected to all neurons in the previous layer, sparsity of connections dictates that a neuron is only connected to specific neurons from the previous layer. This selective connectivity reduces the number of connections in the network, thereby potentially improving computational efficiency and reducing memory requirements.

4.4.1 Dropout Layers

Dropout [16] is a technique that encourages neurons to learn more independent representations by preventing them from relying on specific other neurons. In addition, dropout can be used to create ensembles of different neural networks without incurring a high computational cost. This approach can help prevent overfitting and improve the generalization performance of a model. Ultimately, dropout can be a useful tool for encouraging neural networks to learn more robust representations and improve their performance on a wide range of tasks. It operates as follows:

- Take a value q within 0 and 1, with 0.5 being a commonly used value.
- For each layer, randomly select p fraction of neurons and assign them a weight of 0 such that they are not going to participate in learning.
- During testing, dropout is not utilized, but the output of neurons is multiplied by 1-q, which is the expected proportion of times the neuron was active during training.

4.5 3D Convolutional Neural Network

A 3D Convolutional Neural Network (3D CNN) [12], is a type of neural net specifically built for processing data with spatial and temporal dimensions. While a traditional 2D CNN operates on two-dimensional data, such as images, a 3D CNN extends the concept to three-dimensional data, such as video or volumetric data.

In a 3D CNN, convolutional layers have filters that slide not only spatially across the width and height of the input but also temporally across the depth or time dimension. This let the network capture both spatial and temporal features simultaneously. Basic

building block of a 3D CNN is a 3D convolutional layer, which applies convolutional filters to the input data along all three dimensions. This layer performs convolutions in three dimensions, capturing features that depend on the spatial and temporal relationships in the input. Pooling layers, such as 3D max pooling, can be used to down sample the spatial and temporal dimensions and reduce computational complexity.

3D CNNs are commonly used in tasks that involve video analysis, such as action recognition, video classification, and video segmentation. They are also applied to other types of volumetric data, such as medical imaging, Hyperspectral Imaging and 3D object recognition, where capturing the spatial and temporal dependencies is crucial for accurate analysis.

Overall, 3D CNNs provide a powerful framework for processing and extracting features from three-dimensional data, enabling the network to learn and understand complex spatiotemporal patterns in the input.

4.5.1 Comparisons between 3D Convolutional Neural Network and 2D Convolutional Neural Network

A 3D-CNN and a 2D-CNN differ in their handling of input data and the number of dimensions they operate on. Here are the key distinctions between the two:

Data Dimensionality: A 2D CNN processes two-dimensional data, typically images represented as height and width (spatial dimensions), while a 3D CNN is designed for three-dimensional data, which includes an additional depth or time dimension.

Convolution Operations: In a 2D CNN, convolutional filters slide across the spatial dimensions (width and height) of the input data. The filters perform convolutions in 2D, capturing spatial features. On the other hand, a 3D CNN extends this concept to three dimensions, performing convolutions along the spatial dimensions (width, height) as well as the temporal or depth dimension.

Architecture: Due to the additional dimension, a 3D CNN architecture is more complex than a 2D CNN. A 3D CNN typically includes 3D convolutional layers that operate on the spatial and temporal dimensions, whereas a 2D CNN primarily comprises 2D convolutional layers.

Applications: 2D convolutional neural networks (CNNs) are commonly utilized in various tasks like image classification, object detection, and image segmentation. They are particularly effective in capturing spatial features present in images. On the other hand, 3D CNNs are primarily employed in video analysis tasks such as action recognition, video classification, and video segmentation. These networks are designed to capture both spatial and temporal dependencies, allowing them to excel in capturing spatiotemporal features.

The use of 2D CNNs is beneficial when dealing with static images, as they can effectively extract spatial features by convolving filters over the image grid. However, when analysing videos or sequences of frames, 3D CNNs are more suitable as they can consider the temporal dimension in addition to spatial information. By incorporating temporal dependencies, 3D CNNs can effectively model the changes and dynamics within the video data.

Therefore, while 2D CNNs are suitable for tasks involving static images and spatial analysis, 3D CNNs are specifically designed for video analysis, where spatiotemporal features play a crucial role in understanding the dynamics and motion patterns within the video data.

Input Shape: A 2D CNN expects input data with dimensions (height, width, channels), while a 3D CNN accepts input data with dimensions (depth, height, width, channels).

The additional depth or time dimension allows 3D CNNs to model the temporal changes or volumetric properties in the input data.

In summary, the main differences between 3D CNNs and 2D CNNs lie in the dimensionality of the data they process, the convolution operations they perform, their architectural complexity, the specific applications they are used for, and the input shapes they handle. 3D CNNs are designed to capture spatial and temporal features in three-dimensional data, while 2D CNNs focus on spatial features in two-dimensional data, primarily images.

4.5.2 Hyperspectral Image Classification using 3D Convolutional Neural Network:

The application of 3D CNNs in hyperspectral image (HSI) classification has significantly advanced the field. HSIs pose unique challenges due to the complex acquisition process and limited availability of training samples, which affects classification performance. Unlike traditional CNN-based methods that mainly utilize 2D CNNs for feature extraction, 3D CNNs enable joint spectral-spatial information representation, leading to improved classification accuracy.

By leveraging the spectral dimension, a 3D CNN captures both spatial and spectral features within the HSI data. This comprehensive approach allows for the effective utilization of interband correlations and enhances the discrimination capability of the network. The 3D convolutional layers in the network perform convolutions across the spatial dimensions (width and height) as well as the spectral dimension (depth). This way, the network can capture and model the complex relationships and patterns present in hyperspectral data.

The architecture of a 3D CNN for HSI classification typically consists of multiple 3D convolutional layers, followed by pooling layers for spatial and spectral down sampling. These layers help reduce computational complexity and extract essential features.

Additionally, fully connected layers and a SoftMax layer are incorporated to generate the final classification results.

To further enhance the classification performance, techniques such as transfer learning, regularization, and data augmentation can be applied in conjunction with the 3D CNN architecture. Transfer learning allows the network to benefit from pre-trained models on large-scale datasets, reducing the need for extensive training data. Regularization techniques, such as dropout or batch normalization, help prevent overfitting and improve generalization. Data augmentation techniques, such as random rotations or translations, increase the diversity of the training data and enhance the network's ability to handle variations in the HSI data.

Overall, the integration of 3D CNNs in HSI classification provides a powerful tool for effectively capturing and utilizing the joint spectral-spatial information present in hyperspectral data. This approach improves classification performance by taking advantage of the interband correlations and complex patterns inherent in HSIs.

Chapter 5

Auto Encoders:

Machine learning aims to infer from data automatically without explicit programming. However, for many practical tasks, the labelled data is either small or absent. In such scenarios, unsupervised learning using unlabelled data can help to learn the underlying properties of the features. Autoencoders (AEs) are a type of unsupervised learning approach to learning such properties. AEs can be used for tasks like classification, clustering along with regression. AEs are efficient neural network models that can represent data in a compact and meaningful way. Each data sample is transformed by an AE across several layers of non-linearity, independent of the class information, and trained in an unsupervised manner. The AE is designed to reconstruct the input while passing it through a bottle-neck layer (or layers). In other words, if the input is X, then the output Y = X.

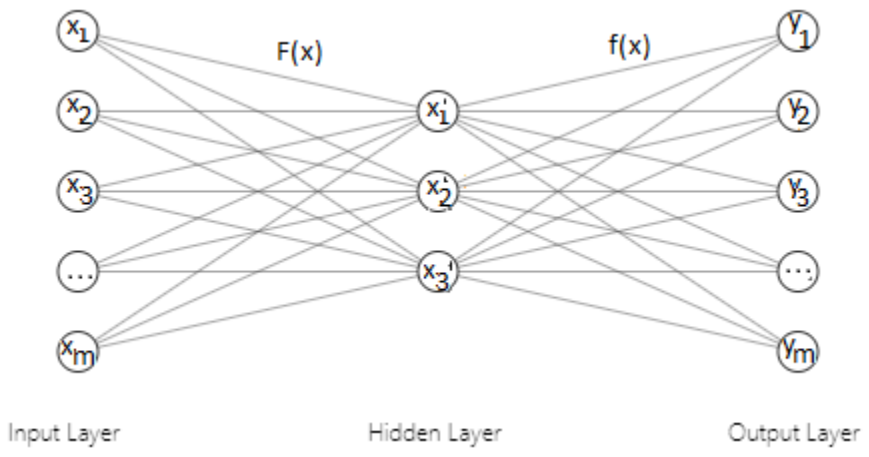


Fig 5.1: Auto Encoder

Training an autoencoder follows the same standard method as any other machine learning neural network, which is through backpropagation. However, there are other training methods available such as extreme learning machines (ELMs) and evolutionary

algorithms (EAs). The main objective during training is to minimize the error of reconstruction, which is calculated using the squared difference between the output and the input [($||\mathbf{y} - \mathbf{x}||^2$)]. Since the output is expected to be identical to the input, the training process is also known as self-supervised learning.

5.1 Different Types of Auto Encoders

There are different types of AEs, which are classified depending on the number of layers, the number of hidden-nodes, and the types of hidden layers. Each type has its strengths and weaknesses. In the following sections, the distinct types of AEs will be described in detail.

It has been argued that if the number of nodes in the hidden layer of a neural network is equal to or greater than the number of nodes in the input layer, the network will only learn the identity function and will not be able to effectively learn underlying features. However, autoencoders are designed to approximate the identity function while passing through a bottleneck, allowing the middle code layer to theoretically learn enough features to reconstruct the input. While the identity function is easy to learn, imposing restrictions on the network, such as limiting the number of nodes in the hidden layers or enforcing sparsity, can lead to the discovery of interesting features. Autoencoders can be categorized as either under-complete or over-complete based on the number of nodes in the hidden layer relative to the input layer.

5.1.1 Under-Complete Auto Encoder

In an autoencoder network, input data is first compressed or encoded to fit into a smaller representation and then decoded to reconstruct the original input. The goal of training is to minimize the reconstruction error, which involves finding the most efficient compact representation or encoding of the input data by passing it through a bottleneck in the network.

5.1.2 Over-Complete Auto Encoder

In a sparse autoencoder network, input data is first transformed or encoded to expand into a much larger representation before being decoded. By applying a sparsity constraint on the hidden-layer, network has been prevented from knowing the identity function simply. This type of network is designed to keep most of the nodes inactive or dormant, with only a few nodes being active. This allows the autoencoder to selectively activate neurons that contribute to extracting meaningful information from the input data, resulting in the discovery of interesting features in the data. An active node in the network is one whose output value is one, while an inactive node has an output value of zero.

5.1.3 Few generic Auto Encoders

There are several distinct variations of the generic Autoencoder (AE) architecture that have been developed to ensure that the transformed data captures the most important features of the original input. These specialized types of AEs have been designed based on the nature of the data and the complexity of the problem. Here, I will describe some popular AEs that are commonly used.

5.2 Classification of Satellite Image Using Auto Encoders

Autoencoders are commonly employed for dimension reduction. By utilizing under-complete AEs in the bottleneck layer, the data can be efficiently compressed to a lower dimensionality. The encoder part of the AE projects the original input into a lower-dimensional representation, while the decoder uses this representation to generate a reconstruction of the input. The output of the encoder is a feature vector with reduced dimensionality, which can be utilized for classification or clustering purposes.

Various types of AEs are employed in different application domains, and their architectures can be customized based on specific requirements. AEs can be shallow, consisting of only input and output layers, or deep, with multiple layers. Increasing the architectural complexity of an AE allows for the learning of more intricate latent representations. For data in grid formats such as images, the encoder and decoder can

be enhanced using convolution and pooling layers. This enables the AE to capture spatial information effectively.

The unique characteristics of AEs can be applied to detect meaningful features in hyperspectral images (HSIs) [2]. Binary change detection in bi-temporal co-registered HSIs poses challenges because of the huge number of bands (Spectral) present in the data. To address this, researchers often employ dimensionality reduction techniques. Additionally, manually labelling pixels in HSIs becomes difficult due to the high number of spectral bands. Unsupervised methods [3] that focus on detecting dissimilarities between the two images can be used without requiring manual labelling [4]. While unsupervised methods are generally less accurate than supervised [1, 2, 5] or semi-supervised [6, 7, 8] approaches, they offer greater sophistication in application. To tackle this problem, a solution can be proposed using a modified convolutional autoencoder (CAE) that preserves both spatial and spectral information in its hidden code layer for effective dimensionality reduction.

Chapter 6

Proposed Methodology:

One of the major concerns of image processing is image recognition. Objects are represented as a collection of pixels in an image. Our task is to describe the region based on the chosen representation.

6.1 Dense Neural Network

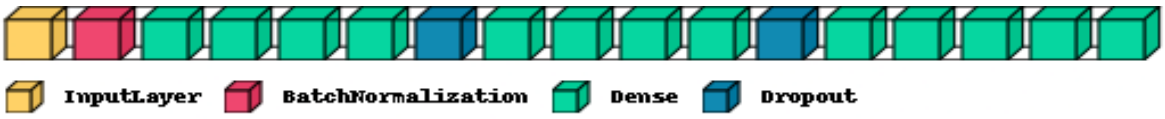


Fig 6.1: Dense Neural Network Architecture

The dense network consists of several fully connected layers. It begins with an input layer of dimensions matching the train sample size, followed by a batch normalization layer. Four dense layers are then added sequentially. A dropout layer with a dropout rate of 0.2 is introduced after these initial dense layers.

Next, an additional four dense layers are added, followed by another dropout layer with a dropout rate of 0.2. Subsequently, four more dense layers are included in the network. The final dense layer serves as the output layer, containing 'n' neurons corresponding to the number of classes.

6.2 Auto Encoder + k-nearest neighbors Architecture

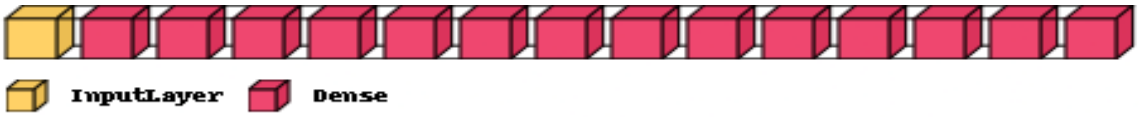


Fig 6.2: Auto Encoder Architecture

In this architecture, the initial six dense layers serve as Encode Layers. The subsequent dense layer represents the code layer. Following that, the next six layers function as decode layers. Finally, the final dense layer works as the output layer for the desired results. This process involves reconstructing the image and removing any existing noises. For the classification task, a K Nearest Neighbor (KNN)[14] Classifier was utilized.

6.3 3D Convolutional Neural Network Architecture

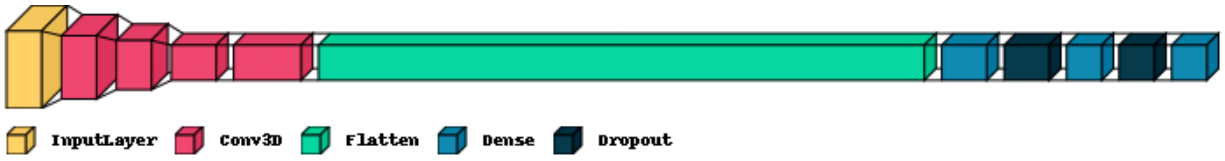


Fig 6.3 : 3D CNN Architecture

In a 3D CNN-based task, a machine with significant computational power is required, which can be limiting for our target. To address this, the dataset was initially subjected to PCA to extract the most important principal components for the task.

For the subsequent 3D convolution task, an input layer with dimensions of (25, 25, 16, 1) was employed. This was followed by a 3D convolution layer with ReLU[21] activation, utilizing 8 filters and a kernel size of (3, 3, 7). Another 3D convolution layer with ReLU activation was added, employing 16 filters and a kernel size of (3, 3, 5). Subsequently, a 3D convolution layer with ReLU activation was introduced, utilizing 16 filters and a kernel size of (3, 3, 3).

To facilitate a 2D convolution operation, a reshape layer was introduced. Following this, a 2D Convolution layer with ReLU activation was added, employing 64 filters and a kernel size of (3, 3). A flatten layer was then included, and subsequently, a dense layer with 256 neurons was added. To prevent overfitting, a dropout layer with a dropout rate of 0.4 was employed. Finally, a dense layer with N neurons (where N represents the number of classes) was added to produce the desired output.

6.4 Accuracy Measurement Technique

Validation accuracy is a crucial metric used to evaluate how well a model generalizes to unseen data during training. It is calculated by assessing the model's performance on a validation dataset that is distinct from the training data. The validation accuracy provides an estimate of how the model will perform on new, unseen data by measuring the percentage of correctly classified examples in the validation dataset.

The actual value of the validation accuracy depends on various factors, such as the complexity of the model, the size and quality of the training dataset, and the choice of hyperparameters. A higher validation accuracy generally indicates that the model is performing better on the validation dataset and is likely to generalize well to new data. However, it is important to note that achieving high validation accuracy does not always guarantee good performance on all types of data and in all contexts.

The mathematical expression for validation accuracy is a formula used to calculate the percentage of correctly classified examples in a validation dataset during model training via true positive and false positive [20].

$$Accuracy = \frac{True\ Positive + False\ Positive}{True\ Positive + False\ Positive + True\ Negative + False\ Negative}$$

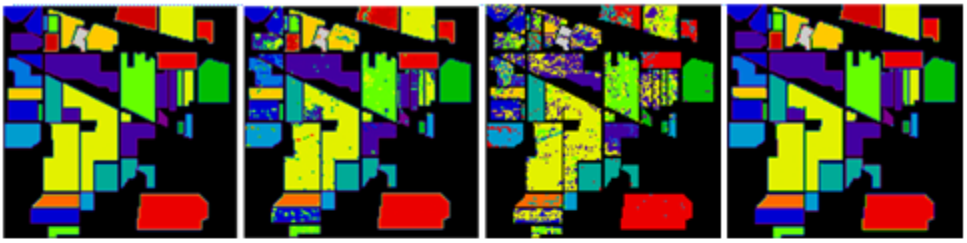
Chapter 7

Experimental Results and Analysis:

The models mentioned were executed in parallel, and the validation accuracy was measured. The classification maps, along with the corresponding ground truth, are displayed below.

7.1 Classified Output of Indian Pines Dataset in different models

Indian Pines dataset has 16 different classes “Alfalfa”, “Corn-notill”, etc. In This portion we classified the classes in different models Dense Neural Network, Auto Encoder based model and 3D CNN based model. Within these models 3D CNN based model performed the best , followed by Dense Network Model and AE based model.

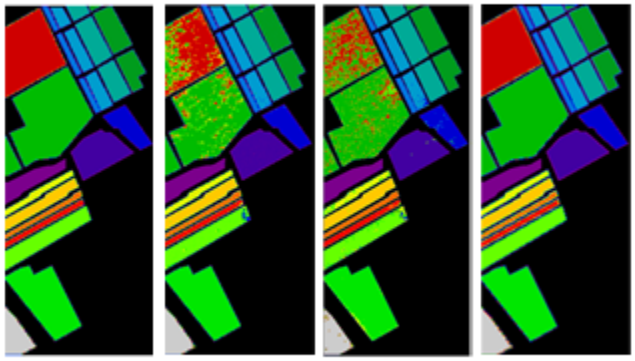


#	Class	Samples	Color
1	Alfalfa	46	
2	Corn-notill	1428	
3	Corn-mintill	830	
4	Corn	237	
5	Grass-pasture	483	
6	Grass-trees	730	
7	Grass-pasture-mowed	28	
8	Hay-windrowed	478	
9	Oats	20	
10	Soybean-notill	972	
11	Soybean-mintill	2455	
12	Soybean-clean	593	
13	Wheat	205	
14	Woods	1265	
15	Buildings-Grass-Trees-Drives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

Fig 7.1: (From the left) The ground truth data, the classification map after using the Dense Neural Network, the classified output after using the Auto Encoder Based Network, and the classification map after using the 3D Convolutional Neural Network.

7.2 Classified Output of Salinas Dataset in different models

Salinas dataset has 16 different classes “Brocoli green weed 1”, “Brocoli green weeds 2”, etc. In This portion we classified the classes in different models Dense Neural Network, Auto Encoder based model and 3D CNN based model. Within these models 3D CNN based model performed the best, followed by Dense Network Model and AE based model.



#	Class	Samples	Color
1	Brocoli_green_weeds_1	2009	
2	Brocoli_green_weeds_2	3726	
3	Fallow	1976	
4	Fallow_rough_plow	1394	
5	Fallow_smooth	2678	
6	Stubble	3959	
7	Celery	3579	
8	Grapes_untrained	11271	
9	Soil_vinyard_develop	6203	
10	Corn_senesced_green_weeds	3278	
11	Lettuce_romaine_4wk	1068	
12	Lettuce_romaine_5wk	1927	
13	Lettuce_romaine_6wk	916	
14	Lettuce_romaine_7wk	1070	
15	Vinyard_untrained	7268	
16	Vinyard_vertical_trellis	1807	

Fig 7.2 : The ground truth data, the classification map after using the Dense Neural Network, the classification map after using the Auto Encoder + KNN Classifier Based Network, and the classified output after using the 3D Convolutional Neural Network.

7.3 Classified Output of Samson Dataset in different models

Samson dataset has 3 different classes “Soil”, “Tree”, and “Water”. In This portion we classified the classes in different models Dense Neural Network, Auto Encoder based model and 3D CNN based model. Within these models all the models performed almost same with a good accuracy.

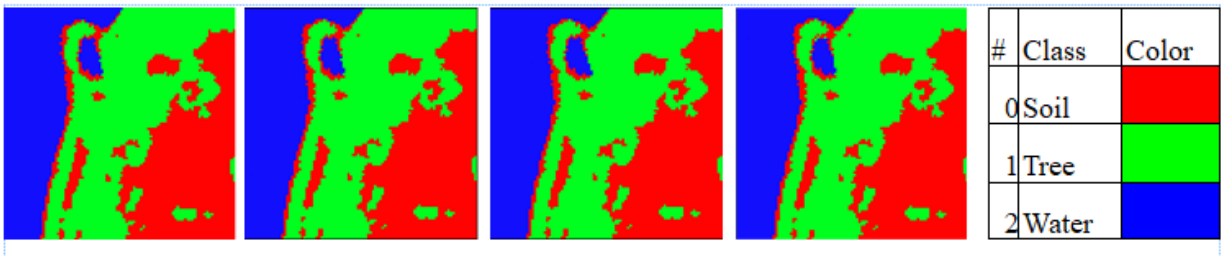


Fig 7.3: The ground truth data, the classification map after using the Dense Neural Network, the classification map after using the Auto Encoder Based Network, and the classified output after using the 3D Convolutional Neural Network.

7.4 Classified Output of Jasper Ridge Dataset in different models

Jasper Ridge dataset has 4 different classes “Tree”, “Soil”, “Water” and “Road”. In This portion we classified the classes in different models Dense Neural Network, Auto Encoder based model and 3D CNN based model. Within these models all the models performed almost same with a good accuracy.

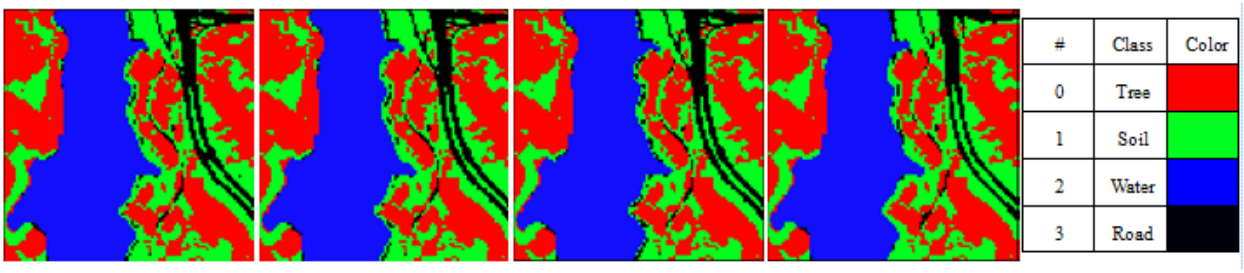


Fig 7.4 : The ground truth data, the classification map after using the Dense Neural Network, the classification map after using the Auto Encoder + KNN Classifier Based Network, and the classified output after using the 3D Convolutional Neural Network.

7.5 Classified Output of Henry Island in different models

Salinas dataset has 6 different classes “End Member 1”, “End Member 2”, …., “End Member 6”. In This portion we classified the classes in different models Dense Neural Network, Auto Encoder based model and 3D CNN based model. Within these models, 3D CNN based model performed the best, followed by Dense Network Model and AE based model.

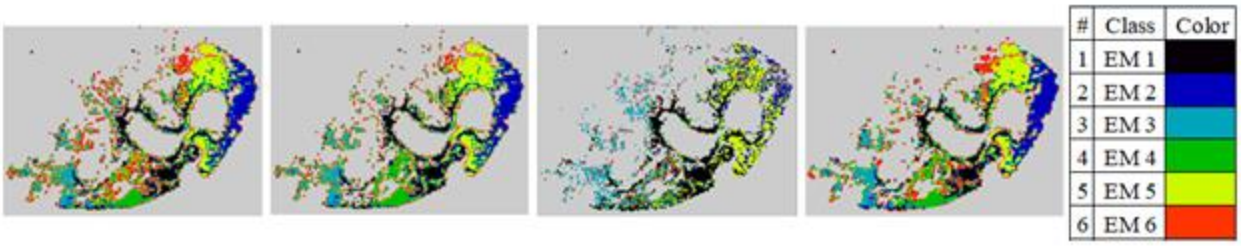


Fig 7.5: The ground truth data, the classification map after using the Dense Neural Network, the classification map after using the Auto Encoder Based Network, and the classified output after using the 3D Convolutional Neural Network.

7.6 Comparison in Validation Accuracy of the Models

The datasets underwent analysis using few separate Deep Learning models: A Dense Neural Network, An Auto Encoder and a 3d CNN model. The obtained results are presented in the following table.

Data Set	Simple Dense Net	Auto Encoder	3D CNN
Indian Pines	87.87	83.95	99.58
Salinas	92.7	94.34	99.51
Samson	99.75	99.83	99.56
Jasper Ridge	97.75	98.58	99.08
Henry Island	95.21	88.04	99.71

Table 7.1: Comparison of Several Deep Learning Models

Chapter 8

Conclusion and Future Work:

3D Convolutional Neural Networks (CNNs) have demonstrated significant efficacy in classifying hyperspectral images. When applied to datasets like Indian Pines and Salinas, which contain numerous classes, alternative methods such as autoencoder + KNN Classifier and simple dense networks experience a decline in accuracy. This decrease can be attributed to the increased complexity of the classification task resulting from the larger number of classes.

The Indian Pines and Salinas datasets encompass various land cover categories, including vegetation, buildings, and water bodies, posing a challenging classification task. Simpler methods like autoencoder + KNN Classifier and simple dense networks may struggle to accurately classify such a diverse range of classes. Consequently, their accuracy performance is likely to be lower compared to more sophisticated techniques like 3D CNNs.

On the other hand, the Samson Dataset comprises a smaller number of classes, simplifying the classification task. With fewer classes, distinguishing between different land cover types becomes less complex. Therefore, the accuracy of all models, including autoencoder + KNN Classifier and simple dense networks, remains relatively consistent across various models. In this scenario, the simpler models can achieve comparable accuracy to more advanced methods, as the classification task is less demanding.

To summarize, when dealing with hyperspectral datasets with a high number of classes, such as Indian Pines and Salinas, 3D CNNs tend to outperform simpler methods like autoencoder + KNN Classifier and simple dense networks due to their capability to capture complex spatial information. However, in datasets with a lower number of

classes like the Samson Dataset, the accuracy of all models remains similar, making the simpler models a viable option.

Future work in this topic could focus on exploring novel architectures for 3D CNNs, developing hybrid models combining different techniques, and investigating transfer learning approaches for hyperspectral image classification.

Chapter 9
References:

1) Y. Zhan, K. Fu, M. Yan, X. Sun, H. Wang, and X. Qiu, “Change Detection based on Deep Siamese Convolutional Network for Optical Aerial Images,” IEEE Geoscience and Remote Sensing Letters, vol. 14, no. 10, pp. 1845–1849, 2017

2) C. Zhao, H. Cheng, and S. Feng, “A Spectral-Spatial Change Detection Method Based on Simplified 3-D Convolutional Autoencoder for Multitemporal Hyperspectral Images,” IEEE Geoscience and Remote Sensing Letters, 2021.

3) S. Liu, L. Bruzzone, F. Bovolo, and P. Du, “Hierarchical Unsupervised Change Detection in Multitemporal Hyperspectral Images,” IEEE Transactions on Geoscience and Remote Sensing, vol. 53, no. 1, pp. 244–260, 2014

4) Y. K. Han, A. Chang, S. K. Choi, H. Park, and J. Choi, “An Unsupervised Algorithm for Change Detection in Hyperspectral Remote Sensing Data using Synthetically Fused Images and Derivative Spectral Profiles,” Journal of Sensors, vol. 2017,2017.

5) Q. Wang, Z. Yuan, Q. Du, and X. Li, “GETNET: A General End-to-end 2-d CNN Framework for Hyperspectral Image Change Detection,” IEEE Transactions on Geoscience and Remote Sensing, vol. 57, no. 1, pp. 3–13, 2018.

6) H. Wu and S. Prasad, “Semi-supervised Deep Learning using Pseudo Labels for Hyperspectral Image Classification,” IEEE Transactions on Image Processing, vol. 27,no. 3, pp. 1259–1270, 2017.

7) F. De Morsier, D. Tuia, M. Borgeaud, V. Gass, and J. P. Thiran, “Semi-supervised Novelty Detection using SVM Entire Solution Path,” IEEE Transactions on Geoscience and Remote Sensing, vol. 51, no. 4, pp. 1939–1950, 2013.

8) M. Roy, S. Ghosh, and A. Ghosh, “A Neural Approach under Active Learning Mode for Change Detection in Remotely Sensed Images,” IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, vol. 7, no. 4, pp. 1200–1206,2013

9) I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning, ser. Adaptive Computation and Machine Learning Series. [Online]. Available: https://books.google.co.in/books?id=Np9SDQAAQBAJ

10) Chakravortty, Somdatta & Ghosh, Dipanwita & Sinha, Devadatta. (2018). A Dynamic Model to Recognize Changes in Mangrove Species in Sunderban Delta Using Hyperspectral Image Analysis. 10.1007/978-981-10-3373-5_5.

11) Rathore, Muhammad Mazhar & Ahmad, Awais & Paul, Anand. (2016). Real-time Continuous Feature Extraction in Large Size Satellite Images. Journal of Systems Architecture. 64. 10.1016/j.sysarc.2015.11.006.

12) 3D convolutional neural network for machining feature recognition with gradient-based visual explanations from 3D CAD models by Jinwon Lee, Hyunoh Lee & Duhwan Mun [https://rdcu.be/dcvFD]

13) Popescu, Marius-Constantin & Balas, Valentina & Perescu-Popescu, Liliana & Mastorakis, Nikos. (2009). Multilayer perceptron and neural networks. WSEAS Transactions on Circuits and Systems. 8.

14) Guo, Gongde & Wang, Hui & Bell, David & Bi, Yaxin. (2004). KNN Model-Based Approach in Classification.

15) Alzubaidi, L., Zhang, J., Humaidi, A.J. et al. Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. J Big Data 8, 53 (2021). https://doi.org/10.1186/s40537-021-00444-8

16) Garbin, Christian & Zhu, Xingquan & Marques, Oge. (2020). Dropout vs. batch normalization: an empirical study of their impact to deep learning. Multimedia Tools and Applications. 79. 1-39. 10.1007/s11042-019-08453-9.

17) Alzubaidi, L., Zhang, J., Humaidi, A.J. et al. Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. J Big Data 8, 53 (2021). https://doi.org/10.1186/s40537-021-00444-8

18) M. Zhao, S. Shi, J. Chen and N. Dobigeon, "A 3-D-CNN Framework for Hyperspectral Unmixing With Spectral Variability," in IEEE Transactions on

Geoscience and Remote Sensing, vol. 60, pp. 1-14, 2022, Art no. 5521914, doi: 10.1109/TGRS.2022.3141387.

19) Arbib, Michael. (1969). Review of 'Perceptrons: An Introduction to Computational Geometry' (Minsky, M., and Papert, S.; 1969). Information Theory, IEEE Transactions on. 15. 738- 739. 10.1109/TIT.1969.1054388.

20) Eric Conrad, Seth Misenar, Joshua Feldman,Chapter 7 - Domain 7: Security operations,Editor(s): Eric Conrad, Seth Misenar, Joshua Feldman,Eleventh Hour CISSP® (Third Edition),Syngress,2017,Pages 145-183,ISBN 9780128112489, https://doi.org/10.1016/B978-0-12-811248-9.00007-3.

21) Deep Learning using Rectified Linear Units (ReLU),Cornell University, https://doi.org/10.48550/arXiv.1803.08375

22) Laban, Noureldin & Abdel latif, Bassam & Ebied, Hala & Shedeed, Howida & Tolba, Mohamed. (2020). Reduced 3-D Deep Learning Framework for Hyperspectral Image Classification. 10.1007/978-3-030-14118-9_2.

23) Kang, Xudong & Li, Shutao & Fang, Leyuan & Li, Meixiu & Benediktsson, Jon. (2015). Extended Random Walker-Based Classification of Hyperspectral Images. IEEE Transactions on Geoscience and Remote Sensing. 53. 10.1109/TGRS.2014.2319373.

24) Zheng, Pan & Su, Hongjun & Du, Qian. (2021). Sparse and Low-Rank Constrained Tensor Factorization for Hyperspectral Image Unmixing. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing. PP. 1-1. 10.1109/JSTARS.2020.3048820.

25) Wilber, Cynthia. (2008). Field based learning: Outreach education programs at Jasper Ridge Biological Preserve, Stanford University.

26) Ghosh, Dipanwita & Chakravortty, Somdatta. (2022). Reconstruction of High Spectral Resolution Multispectral Image using Dictionary-based Learning and Sparse Coding. Geocarto International. 37. 1-16. 10.1080/10106049.2022.2040601.

27) Sarker, I.H. Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions. SN COMPUT. SCI. 2, 420 (2021). https://doi.org/10.1007/s42979-021-00815-1

28) Bank, Dor & Koenigstein, Noam & Giryes, Raja. (2020). Autoencoders.

29) Guo, Gongde & Wang, Hui & Bell, David & Bi, Yaxin. (2004). KNN Model-Based Approach in Classification.

30) Gelenbe, Erol & Yin, Yonghua. (2017). Deep Learning with Dense Random Neural Networks. 10.1007/978-3-319-67792-7_1.

31) Green, Robert & Chrien, Thomas & Nielson, P. & Sarture, Charles & Eng, Bjorn & Chovit, Christopher & Murray, Alex & Eastwood, Michael & Novack, H.. (1993). Airborne visible/infrared imaging spectrometer (AVIRIS): recent improvements to the sensor and data facility. Proc SPIE. 180-190. 10.1117/12.157056.

32) Mohd Yusof, Ain Zat & Abdul Manap, Redzuan & Darsono, Abdul. (2020). A comparative study of hyperspectral unmixing using different algorithm approaches. Indonesian Journal of Electrical Engineering and Computer Science. 20. 813. 10.11591/ijeecs.v20.i2.pp813-821.