Technische Universität Berlin

Faculty of Electrical Engineering and Computer Science
Dept. of Computer Engineering and Microelectronics
Remote Sensing Image Analysis Group

EXPLAINING AND INTERPRETING CNNS FOR CLASSIFICATION OF HIGH DIMENSIONAL SATELLITE IMAGES

Master of Science in Computer Science

January, 2021

Ananya Bhanja Chowdhury

Matriculation Number: 385932

Supervisor: Prof. Dr. Begüm Demir

Advisor: Dr. Mahdyar Ravanbakhsh

Eidesstattliche Erklärung

Hiermit erkläre ich an Eides statt, dass ich die vorliegende Arbeit selbstständig und ohne fremde Hilfe angefertigt habe. Sämtliche benutzten Informationsquellen sowie das Gedankengut Dritter wurden im Text als solche kenntlich gemacht und im Literaturverzeichnis angeführt. Die Arbeit wurde bisher nicht veröffentlicht und keiner Prüfungsbehörde vorgelegt.

Hereby I declare that I wrote this thesis myself with the help of no more than the mentioned literature and auxiliary means.

Berlin, Date 22.01.2021

. . . . . .

Ananya Bhanja Chowdhury

Acknowledgements

First and foremost I would like to express my gratitude and thanks to my supervisor Prof. Dr. Begüm Demir for her patience and guidance throughout the duration of my thesis. I am very grateful that she granted me the opportunity to work with her department. This was a huge learning opportunity for me and I hope that I will be able to take all the knowledge forward with me for my future endeavours. I would like to express my sincere thanks and gratitude to my advisor Dr. Mahdyar Ravanbakhsh. Without his supervision, guidance, and continued support it would not have been possible to complete my thesis. I would also like to thank my family for their constant support and encouragement, even while being so far away from each other.

/

Abstract

Image classification is one the first steps in most complex remote sensing (RS) tasks and forms the very foundation of activities like land-use/cover study, forest and agriculture management, urban planning etc. Effective classification of remote sensing images is of high importance as it can help in deciphering the core components classes in each image. Deep learning (DL) has shown remarkable progress in RS and has achieved high degree of success and accuracy in classifying remote sensing images. Convolutional neural network (CNN) is one of the most successful deep learning tools that is used for image classification. The choice of CNN architecture has a big impact on the overall classification performance. The most important component in a CNN is the convolution layer. This is because, the convolution layer is in charge of transforming the input data across multiple axes, using the multi-dimensional filters. This enables it to identify basic shapes and structures belonging to classes in the image. Thus, different convolution types, like 2D convolution and 3D convolution, and parameters like dilation can have a significant influence on the feature extraction process and overall performance of the CNN. This master thesis intends to study these effects and for this purpose proposes to use different types of convolution (2D,3D convolutions) in combination with hyperparameters (dilation) to create multiple CNNs. In this study, these networks has been be used to study the effect of each kind of convolution parameters in learning and identifying features belonging to different RS classes. This is crucial in determining the appropriate architecture for RS images, especially for multi-spectral multi-labelled images. This master thesis uses multi-labelled images from the BigEarthNet [1] remote sensing archive, which are annotated with 19 different land-cover classes. The CNNs designed in this master thesis uses the k-branch [2] architecture. These CNNs have been used to classify the images from BigEarthNet archive. Based on the classification results, comparative analysis of the architectures are performed. This is important in order to understand the features or classes that can be extracted better using 2D convolution, 3D convolution and dilation. To this end, different visualization techniques like visualization of feature maps, Layer-wise Relevance Propagation (LRP) and Gradient-weighted Class Activation Mapping (Grad-CAM) have been used to evaluate the performance of the network architectures. These techniques give significant insight into the decision-making process of a CNN by highlighting the predicted class and areas with highest influence on the classification result. In this thesis, these visualization techniques showed distinct improvement from 2D to 3D convolution in localizing areas belonging to a predicted class in an image. The goal of this master thesis is to provide valuable insights into the architectural specifications that influence the classification results and thus, make the classification process transparent, verifiable and reliable.

Zusammenfassung

Die Bildklassifizierung ist einer der ersten Schritte bei den meisten komplexen Fernerkundungsaufgaben und bildet die Grundlage für Aktivitäten wie Studien zur Landnutzung und -bedeckung, Forst- und Landwirtschaftsmanagement, Stadtplanung usw. Eine effektive Klassifizierung von Fernerkundungsbildern ist von großer Relevanz, da sie bei der Entschlüsselung der Kernkomponenten-Klassen in jedem Bild helfen kann. Deep Learning (DL) hat in der RS große Fortschritte gemacht und einen hohen Grad an Erfolg und Präzision bei der Klassifizierung von Fernerkundungsbildern erreicht. Das Convolutional Neural Network (CNN) ist eines der erfolgreichsten Deep-Learning-Tools, das für die Bildklassifizierung verwendet wird. Die Wahl der CNN-Architektur hat einen großen Einfluss auf die Gesamtklassifizierungsleistung. Die wichtigste Komponente in einem CNN ist die Faltungsschicht. Das liegt daran, dass die Faltungsschicht für die Transformation der Eingabedaten über mehrere Achsen zuständig ist, indem sie die mehrdimensionalen Filter verwendet. Dadurch ist sie in der Lage, grundlegende Formen und Strukturen, die zu Klassen gehören, im Bild zu erkennen. So können verschiedene Faltungstypen, wie 2D-Faltung und 3D-Faltung, und Parameter wie Dilatation einen signifikanten Einfluss auf den Feature-Extraktionsprozess und die Gesamtleistung des CNNs haben. Diese Masterarbeit beabsichtigt, diese Effekte zu untersuchen und schlägt zu diesem Zweck vor, verschiedene Arten von Faltung (2D-, 3D-Faltung) in Kombination mit Hyperparametern (Dilatation) zu verwenden, um mehrere CNNs zu erstellen. In dieser Studie wurden diese Netzwerke verwendet, um den Effekt jeder Art von Faltungsparametern beim Lernen und Identifizieren von Merkmalen, die zu verschiedenen RS-Klassen gehören, zu untersuchen. Dies ist entscheidend für die Bestimmung der geeigneten Architektur für RS-Bilder, insbesondere für multispektrale, mehrfach gelabelte Bilder. Diese Masterarbeit verwendet mehrfach gelabelte Bilder aus dem BigEarthNet [1] Fernerkundungsarchiv, die mit 19 verschiedenen Landbedeckungsklassen annotiert sind. Die CNNs, die in dieser Masterarbeit entwickelt wurden, verwenden die k-branch [2] Architektur. Diese CNNs wurden benutzt, um die Bilder aus dem BigEarthNet-Archiv zu klassifizieren. Basierend auf den Klassifikationsergebnissen wird eine vergleichende Analyse der Architekturen durchgeführt. Dies ist wichtig, um zu verstehen, welche Merkmale oder Klassen mit 2D-Faltung, 3D-Faltung und Dilatation besser extrahiert werden können. Zu diesem Zweck wurden verschiedene Visualisierungstechniken wie die Visualisierung von Feature-Maps, Layer-wise Relevance Propagation (LRP) und Gradient-weighted Class Activation Mapping (Grad-CAM) verwendet, um die Leistung der Netzwerkarchitekturen zu bewerten. Diese Techniken geben einen signifikanten Einblick in den Entscheidungsprozess eines CNN, indem sie die vorausgesagte Klasse und die Bereiche mit dem größten Einfluss auf das Klassifikationsergebnis hervorheben. In dieser Arbeit zeigten diese Visualisierungstechniken eine deutliche Verbesserung von der 2D- zur 3D-Faltung bei der Lokalisierung von Bereichen, die zu einer vorhergesagten Klasse in einem Bild gehören. Das Ziel dieser Masterarbeit ist es, wertvolle Einblicke in die architektonischen Vorgaben, die

das Klassifikationsergebnis beeinflussen, zu geben und damit den Klassifikationsprozess transparent, überprüfbar und zuverlässig zu machen.

Contents

List of Acronyms

CNN	Convolutional Neural network
DL	Deep Learning
RS	Remote Sensing
HSI	Hyperspectral Image
MSI	Multispectral image
RGB	Red Green Blue
NDVI	Normalised Difference Vegetation Index
RVI	Ratio Vegetation Index
2D	2 Dimensional
3D	3 Dimentional
CAM	Class Activation Map
GRAD-CAM	Gradient-weighted Class Activation Mapping
LRP	Layer-wise Relevance Propagation
XAI	Explainable Artificial Intelligence
GAP	Global Average Pooling
IR	Infra Red
NIR	Near Infrared
SWIR	Short Wave Infrared
VNIR	Very Near Infrared
ReLU	Rectified Linear Unit
SVM	Support Vector Machines
RF	Random Forest
HPC	High performance Computing
TP	True Positive
FP	False Positive
TN	True Negative
FN	False Negative

List of Figures

List of Tables

1 Introduction

Remote Sensing (RS) Image classification is an area of science where deep learning (DL) has been successfully and extensively used over the last few years. With the improvement in image acquisition capabilities of low-earth satellites, it is now possible to acquire large number of high quality images. Satellite images are different from aerial photography in a sense, that satellite images are comprised of image patches and contain multiple spectral bands of discrete wavelengths. These discrete wavelengths include for example: red, green, blue bands from the visible spectrum of light, as well as infra-red, near infra-red waves, which are invisible to the human eyes. The spatial and spectral dimensions of such images also vary depending on the instruments and sensors in use. Satellite images are created by sensors that capture the light reflected off the surface of the earth. They can be either Multi-spectral images (MSI) or Hyper-spectral images (HSI). These images provides a huge amount of information about the surface of the earth, the land-forms and their characteristics. These information help in land-use/land-cover study, identification of seasonal variations of forest vegetation or agriculture patterns, forest fire damage assessment etc. All these functionalities use image classification as the basic step to identify the features contained and correctly annotate the classes to which they belong. Since remote sensing images are representations of earth's land-forms, they mostly comprise of multiple classes co-existing in a single image. This increases the complexity and challenge of the task. Multi-label scene classification thus forms the very foundation of research in remote sensing using DL.

1.1 Motivation

Convolutional Neural Networks (CNN) is one of the most powerful tools in DL that has been highly successful in image classification and object detection tasks. A CNN comprises of neurons arranged in hierarchical layers, that can identify underlying features in the input information and help in deciphering the object of interest. CNN is a category of neural network and is a successful and effective tool for both classification and regression scenarios. It is heavily used for image classification across domains using RGB images, aerial images, remote sensing images, medical images with very high performance. The papers [3], [4], [5], [6], [7], [8] provide a few examples of CNN being used across domains to perform image classification tasks.

However, despite their success, machine learning models are often known as "black boxes". This is because machine learning models cannot provide any explanation for their result. This makes the process by which a machine learning model reaches a conclusion, completely hidden to its human users. This thesis tries to look inside a CNN using remote sensing image data and analyse the impact of different convolution and explain the efficacy of each convolution by looking into the characteristics of the data that influenced the CNN's decision.

1.2 Objective

The study conducted here is an exploratory analysis of the different convolution parameters and their impact on classification results. The motivation is to understand the changes in classification behavior caused by changes in convolutional parameters for land-cover classes comprising of multi-band patches. Multiple studies have already demonstrated the higher degree of success achieved by using specific convolution, for example 3D convolution, Atrous convolution, parameters compared to normal CNN at handling specific use-cases.

In the paper [9] Li et al. have used a dilated convolutional neural network as a back-end network to generate superior density maps for crowded scenes which enabled them to extract improved saliency information from the data while keeping the image resolution intact. In the paper [10] authors Cui et al. proposed an innovative architecture making use of a combination of residual connections, dilations, strides and 1x1 convolution, and were able to improve the features extracted from the fused features based on hyper-spectral images. Ji et al. in the paper [11] demonstrated the use of 3D Convolution to achieve improved crop classification based on spatial-temporal remote sensing images. Although the improvements caused by convolutional parameters and different architectural techniques are clearly depicted in the respective papers, the impact of these changes on the model's internal perception of the inputs features are unclear. This is the area that this master thesis seeks to address. It tries to understand how addition of padding, dilation or use of a different convolution, changes the perception of the input in a model. Additionally, this study also tries to find a best fit amongst these architectural changes and different satellite bands and image classes.

In order to explain the impact of the above discussed parameters, different techniques of explainable artificial intelligence (XAI) have been used in this thesis. XAI provides methods and techniques to explain the strategy used by a model to draw inference, understand internal representation of input or identify areas of interest in the input. Thus, this also helps in eliminating mistakes caused by influences of spurious correlations [12]. These strategies can be grouped under different categories, for example gradient based analysis, local perturbation based analysis,and propagation based analysis. These techniques provide means to visualize the internal representation of a neural network's perceptions and explain the conclusion reached by it. Heatmap is a popular mode of visualization employed in XAI. It highlights the object of interest, area of maximum importance or localizes the feature discerned by the model. For this thesis, two popular methods, namely layer-wise relevance propagation (LRP) [13] and Grad-CAM[14], have been used to explain the classification choices.

For conducting the experiments under this thesis, the BigEarthNet [1] dataset has been used. BigEarthNet is a repository comprising of 590,326 multi-spectral image patches from the Sentinel-2 satellite system. A basic architecture has been defined and used as a baseline for the results. Next, multiple variations of this architecture, using different dilation parameters and convolutions, have been used to conduct the experiments. Afterwards, the results from each model is analysed and selected candidates out of those are visualized using XAI tools.

1.3 Outline

The remaining thesis has been structured into a total of five chapters that discuss specific areas of the work. A brief description of these chapters are as follows:

Chapter 2 is the **background and related work** section. This section is dedicated to discussing the fundamental background concepts of remote sensing imagery, convolutional neural network and XAI. The chapter is divided into a few sections as below:

- The first section presents essential background concepts of remote sensing, resolutions and band information in satellite imagery.
- The next section introduces the concept of neural network, discusses the details of convolutional operation, different types of convolutions and hyperparameters and how to visualize internal activations of feature maps of a CNN using. It also discusses application of convolutional neural networks in RS.
- This section briefly describes the concepts of XAI and discusses important state-of-the-art interpretability techniques like, LRP [13], Grad-CAM[14], deconvolution [15].

Chapter 3 This chapter defines the **plan of action** for this thesis. In this chapter, relevant research questions and concepts that need to be tested have been presented. The plans presented in this chapter have been justified based on relevant concepts and scientific papers discussed in chapter 2.

Chapter 4 This chapter describes **experimental setup**, basic architecture of the model, dataset and metrics used for this thesis. It also describes the experiments designed for classification of RS images and the techniques used for visualizing features that were learnt by the models.

Chapter 5 This chapter is for **evaluation of the experiments**. It discusses the classification results obtained from each of the experiments and provides a comparative analysis between the models' performances. Afterwards, the results obtained using different visualization techniques have been presented and compared with respect to the class labels and experimental architectures used to perform them.

Chapter 6 This is the **concluding chapter** and summarizes the thesis by discussing the observations, challenges faced, limitations of the study and the opportunity for future work on this topic.

2 Background and Related Work

In this chapter, some of the concepts necessary to understand the thesis have been discussed. The objective of this thesis is to obtain a deeper understanding of CNNs in remote sensing domain. Therefore, this chapter has been divided into multiple sections and subsections that cover topics like remote sensing (2.1), implementation of machine learning in remote sensing (2.2.8), concept of convolution in CNNs and its different parameters (2.2.6), and essential concepts from XAI (2.3).

2.1 Remote Sensing and Satellite imagery

Remote sensing is defined as the science of acquiring information about an object without any direct contact with it. This is done with the help of sensors that are mounted on satellites, or airplanes, which record the reflected or emitted energy from the surface of the monitored object. This thesis uses remote sensing images of the earth's surface taken using satellites.

The information recorded from these reflected or emitted energies correspond to various wavelengths of the electromagnetic spectrum. Depending on the nature of the information, it can be recorded as either images or other non-image formats, like sonar images [16].

Every kind of object on earth's surface absorbs and reflects back some part of sunlight. Depending on multiple factors like albedo, composition and nature of the object, this reflected energy falls at different ends of the electromagnetic wave spectrum. As is commonly known, the electromagnetic spectrum comprises of different wavelengths. Waves with larger wavelengths have lower frequency and vice versa. The figure 2.1 depicts the different components of the electromagnetic spectrum.

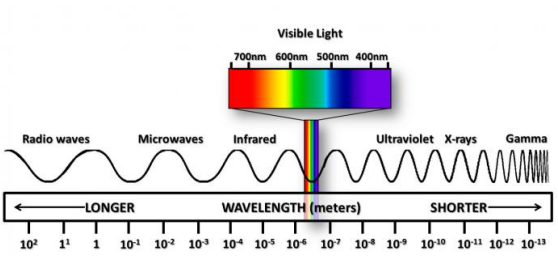


Figure 2.1: Range of Electromagnetic spectrum [17]

Naturally, the waves recorded by the remote sensing sensors range across the spectrum and allow identification of different land cover, vegetation, structures etc. Apart from the visible

spectrum, infra-red (IR), near infra-red (NIR) and shortwave infra-red (SWIR) are also of great interest and use in remote sensing imagery[18].

In the subsequent parts, some important concepts related to satellite imagery, for example, resolution of images and the methods of acquiring them, have been discussed.

2.1.1 Methods of acquiring satellite images

Satellites or airplanes carrying sensors or imaging instruments are used to capture the remote sensing data. There are two types of sensors that can be used to acquire these data. They are: active and passive sensors. [18]

- **Passive sensor:** These are the sensors that depend on the sunlight and record the sunlight reflected off the earth's surface. They are primarily used for detecting waves from visible, very near infrared (VNIR), microwave regions of the spectrum, but are rendered useless by dense cloud cover. Example of passive sensors include: radiometers, spectrometers etc[18].

- **Active sensors:** These are the sensors that provide their own illumination or source of energy and record the component reflected back to it from the object of interest. They are less susceptible to adverse atmospheric conditions. Example of active sensors include: altimeter, scattermeters, and sensors used for measuring radio waves[18].

2.1.2 Resolution of remote sensing images

This section briefly describes the resolution of images in remote sensing. Resolution is an important characteristic and is essential in understanding the information captured by the sensors. It determines how the data from the sensor can be used and the amount of information contained. Resolution can be of the following four types:

- **Spectral resolution:** This is the measure of the range of wavelengths and number of channels or bands that the sensor on board the satellite can measure. If the satellite carries a multispectral sensor, then from 3 to 10 bands or channels can be measured. For hyperspectral sensors, the number of bands measured can be in hundreds[18].

- **Spatial resolution:** This is the measure of the size of the earth's surface that is represented by the size of each pixel. This means that if the spatial resolution is 1km, then an area of size 1km x 1km on earth's surface is captured by one pixel of the sensor[18]. A smaller spatial resolution indicates that higher precision of details are captured by the satellite. In practice, a range of different spatial resolutions are captured by each satellite. Spatial resolution for Sentinel 2A, 2B satellites are 10m, 20m, 60m[19].

- **Radiometric resolution:** This is the measure of the amount of intensity or reflectance correctly captured by the sensor. The higher the radiometric resolution, the better the accuracy of information, and this means finer details will be recorded by the sensor[20][18][19].

- **Temporal resolution:** This is the revisit time of a satellite over the same area. This helps record the changes of the earth's surface in that frequency of time. It can be as little as 30 mins to 1 min for geostationary satellites and in days for polar satellites. For example, Landsat has a temporal resolution of 16 days, while that of the combined Sentinel satellite constellation is 5 days [18][19].

For this thesis, multispectral satellite images from the BigEarthNet [1] archive has been used. These images were acquired using the Sentinel-2 satellite constellation. The Sentinel-2 satellite constellation comprises of two satellites, that measure MSI over 13 bands. The table 2.1 gives the spatial resolutions for each spectral band of Sentinel satellites.

Sentinel-2 bands	Spatial resolution (m)
Band 1 – Coastal aerosol	60
Band 2 - Blue	10
Band 3 - Green	10
Band 4 - Red	10
Band 5 - Vegetation red edge	20
Band 6 - Vegetation red edge	20
Band 7 - Vegetation red edge	20
Band 8 - NIR	10
Band 8A - Narrow NIR	20
Band 9 - Water vapour	60
Band 10 - SWIR - Cirrus	20
Band 11 - SWIR	20
Band 12 - SWIR	20

Table 2.1: Spectral Bands with spatial resolutions for Sentinel satellites [21]

2.1.3 Satellite bands and image composites

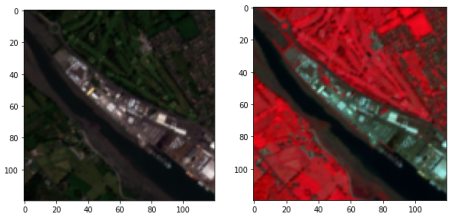
It can be seen from the earlier sections that satellite images capture electromagnetic waves from all across the spectrum. Depending on the spectral and spatial resolution of the data, a lot of information can be learned about the surface of the earth like the land use, condition of vegetation, condition of waterbodies, monitoring of urban areas etc. In fact, objects on earth's surface have their own unique spectral signatures that help in their identification and monitoring. The image bands can be used individually or can be combined together to extract meaningful information. The following table 2.2 gives the order of combination of spectral bands for Sentinel-2 satellites and the purpose such a composite can serve.[22]

Band combination	Purpose
4-3-2	Natural colour, as seen by human eye
8-4-3	False colour (infrared), good for identifying vegetation and waterbodies
12-8-4	False colour (SWIR), good for identifying amount of water content of plants, snow and ice clouds, burnt land
11-8-2	False colour (argiculture), useful for monitoring crop health, water content of plants and soil,snow and ice, burnt land

Table 2.2: Spectral Bands combinations and uses for Sentinel satellites [22]

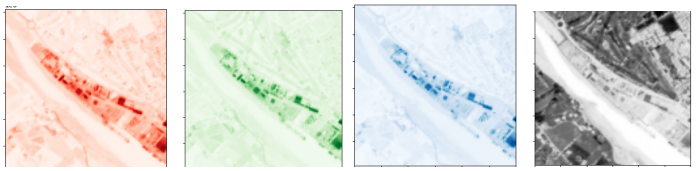
Some other popular composites are Normalised Difference Vegetation Index (NDVI), Ratio Vegetation Index (RVI) [23]. Apart from composites, individual bands can also be very useful in identifying features, for example, NIR band can highlight shores of waterbodies and detect vegetation. Red edge bands can help in classifying vegetation. Band red is useful for identifying vegetation as well as urban areas. Band green can help in identifying oil spill, clear and murky water, band blue helps in soil,vegetation and forest classification[24].

Two examples of raw satellite band and the composite images generated from them have been presented below. Figures 2.2, 2.4 shows the natural (RGB) and False colour composite images created using the band compositions, 4-3-2 and 8-4-3 respectively as given in 2.2 above. The raw bands for the composites are also presented in figures 2.3 and 2.5. It shows the red, green, blue and near infrared bands respectively.



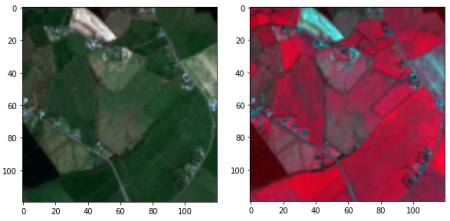
(a) Natural colour image(R-G-B) (b) False colour image

Figure 2.2: Natural and False colour composite images



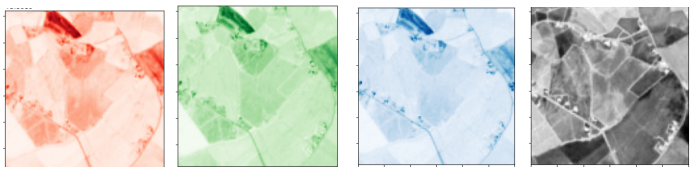
(a) Band Red (b) Band Green (c) Band Blue (d) Band NIR

Figure 2.3: Red, Blue, Green and Near infrared Bands for the same image



(a) Natural colour image(R-G-B) (b) False colour image

Figure 2.4: Natural and False colour composite images



(a) Band Red (b) Band Green (c) Band Blue (d) Band NIR

Figure 2.5: Red, Blue, Green and Near Infrared Bands for the same image

Machine learning has made a lot of contributions to the different areas of application of remote sensing. Research has shown consistent improvement in performance of tasks like classification, change detection, feature identification, vegetation monitoring etc. Further details of application of machine learning techniques in remote sensing has been provided in the section 2.2.8.

2.2 Convolutional Neural Network

This section starts with a brief history of deep learning and neural networks. Then it defines a convolutional neural network (CNN), and introduces important CNN architectures. Next, the

working principle of the convolution operation and impact of its different parameters, like strides and dilation, have been discussed in details, along with a few recent scientific research supporting or demonstrating their advantages.

2.2.1 Emergence of deep learning

The idea, that one day machines will have the capability to think like humans, has always fascinated scientists. With the advancement in computer science, researchers started designing systems that can perform heavy computation and follow complex rules. The domain of machine learning emerged out of these efforts around 1940s and have since undergone many changes resulting in the rise of deep learning around 2006 [25, Chapter 1].

The goal of machine learning has been to let computers learn different ways to perform complex tasks by means of understanding inherent properties of the input data, their desired output, rules and patterns governing the output and learning the rules of transformation, all without human intervention. Such tasks cannot be expressed through hard-coded rules, hence they require the computers to "think" about the rules of the transformation. Deep learning (DL) is a particular branch of machine learning which uses multiple layers of mathematical transforms to process the data and learn its inherent patterns. The system is thus able to learn a hierarchy of information about the input data, with complicated concepts slowly built on top of simpler ones.

Machine learning problems can belong either of four different branches [26, Chapter 4]: (i) Supervised learning; (ii) Unsupervised learning; (iii) Self-supervised learning ; and (iv) Reinforcement learning.

The nature of machine learning problems can be roughly divided into two groups: classification and regression tasks. The task undertaken in this thesis is a classification task with supervised learning. Hence, all forthcoming explanations have been carried out keeping supervised learning and classification in mind.

Machine learning and deep learning has been widely applied across domains to perform complex tasks from identifying handwritten digits, classifying images, identifying facial features to critical tasks like autonomous driving, visual perception tasks, natural language processing. It is implemented across domains including healthcare, banking and insurance sector. In the sub section 2.2.8 of this thesis, concrete examples of application of machine learning in RS tasks have been provided.

2.2.2 Neural networks

It is necessary to have a brief introductory overview on neural networks or feedforward neural networks first, before commencing on to discussing CNNs. Neural networks are one of the most successful deep learning tool.

Neural network can be described as an acyclic graph of neurons. The neurons are arranged in the form of multiple hierarchical layers, known as fully connected layers, where output of one neuron is connected to the input of its next neuron. Figure 2.6 shows the architecture of a typical feedforward network.

The layers seen in figure 2.6 can be divided into input, output and hidden layers. The input layer has the same size as the input data and acts as the interface for the hidden layers. The

ensuing layers until the output layer are known as hidden layers. These hidden layers perform the main data transformation operations. The output layer accumulates the result from all the hidden layer and calculates the loss function to fine-tune the performance of the neural network.

Working principle:

A neural network has to be first trained on data based on the specific problem that it seeks to address. The brilliance of a neural network lies in the fact that it learns to identify salient features of the input data using its interconnected network of neurons and also calibrates its learning process on its own.

Each of the neuron connections seen in figure 2.6 is assigned a weight w_0 at the beginning. When input data (x_0) is provided to the network, each neuron calculates a sum of all the weighted input (w_0x_0), which is basically a matrix multiplication, and then applies an activation function to calculate the unit's output. Activation of a neuron simply indicates whether the neuron got activated or triggered in response to the input data.

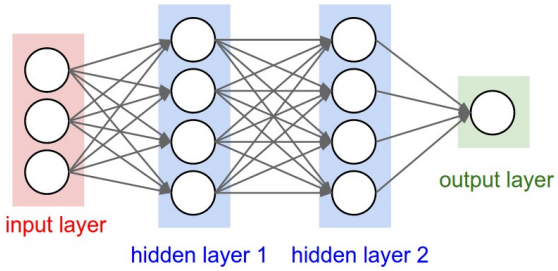


Figure 2.6: neural network architecture [27]

Neural networks use vector representation to characterize input data. Thus, the input data can be written as a vector x=$\{x_0, x_1, ...x_n\}$. The weights of a layer are represented as a two-dimensional matrix W=$\{w_0, w_1, ...w_n\}$ while the biases are denoted by a vector b=$\{b_0, b_1, ...b_n\}$. As already stated, the activation of neurons of a layer are given by summation of the dot product of W and x i.e. $(W.x)$. The hidden layer activation is obtained by applying the activation function f on the total activation[27, Module 1]. The equation 2.1 depicts the hidden layer output activation.

$$f(\sum_i w_i x_i + b) \tag{2.1}$$

No activation function is applied in the output layer. A simple summation of weighted input and bias forms the output of the neural network. Such a simple network can be converted into a classifier by applying a appropriate loss function and optimizer.

Weights and learnable parameters

The complexity of a neural network depends on the number of hidden layers and the number of trainable parameters in the network. In the figure 2.6 the neural network has 2 hidden layers. The number of trainable parameters can be calculated by summation of the number of connections

between (i) input parameters and first hidden layer; (ii) all the hidden layers; (iii) last hidden layer and output parameters; and (iv) by a summation of total number of biases in the network.

Thus for the neural network depicted in figure 2.6, the total number of weights are: (3x4) + (4x4) + (4X1) = 32 and number of bias values are 4 + 4 + 1 = 9. Therefore, there are a total of 41 different trainable parameters associated with this neural network[27, Module 1].

Although 41 trainable parameters seem minor, the number increases exponentially when an image data is used. Neural networks convert input data into vectors. This has a number of disadvantages. These disadvantages can be demonstrated by taking the example of an image as an input to a neural network. A natural colour image with three channels (i.e. RGB image) of dimension 120x120x3 will be thus converted into a vector of 120*120*3 = 43200 units. The first hidden layer will have 43,200 number of weights. This increases significantly with each hidden layer. For larger images the total number of trainable parameters increases drastically. This does not scale well due to heavy constraint in computation, and are often prone to overfitting. These problems can be solved (or mostly addressed) by using a CNN.

2.2.3 Convolutional Neural Networks

A Convolutional neural network(CNN) is a variation of the fully connected neural network. The working principles of a neural network discussed in 2.2.2 largely holds for a CNN as well.

Unlike fully connected networks, CNN uses multiple different kinds of layers, the most important out of which is the convolutional layer. This layer is in-charge of performing convolution operation between the provided input data and the weights of each layer (known as filters in a CNN). The neurons in a CNN are not fully connected to one another. CNNs are primarily used to perform either classification or regression problems. In case of classification, the output layer is of the same dimension as the number of classification classes.

CNNs were constructed by keeping computer visions problems in mind and were adapted to accept images, in the form of matrices, as input. This is advantageous as it preserves the spatial relationship between pixels of the image. Its advantage lies on the fact that neighbouring pixels in an image generally correspond to a single object. A CNN leverages this characteristic of image pixels while selecting the size of the receptive fields.

Although applicable to most different kinds of data, CNNs are particularly useful in solving problems using image data, time series data, medical imaging, audio data, handwritten text, natural language processing etc.

Historical background and important CNN architectures

The first application of convolutional neural networks was demonstrated in 1998 by LeCunn et al. [28]. The paper proposed a model,that became famous as LeNet architecture, to classify handwritten numbers and zip codes. Since then, progressively more complex architectures have been proposed and has proved successful in performing highly complex tasks. In 2012, the AlexNet architecture using deep CNN won the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) for Image classification using large-scale image database [29]. This was an important breakthrough and had a profound impact on the computer vision domain. Since then, many complex CNN architectures have come forth like VGG-16 [30], Inception [31], ResNet

[32], Xception [33]. These complex architectures introduced very deep CNNs models (up to 152 layers deep) and popularized techniques like using Rectified Linear Unit (ReLU), Batch Normalization, 1x1 Convolution, shortcut connections, residual blocks and revolutionized image classification.

2.2.4 Basics of CNN architecture

As discussed in the above section, a CNN is essentially a stack of different layers, like, convolutional layer, fully connected layer, pooling layer, batch normalization and dropout layer. Figure 2.7 illustrates the architecture of a simple CNN. This section briefly discusses the important layers of CNN. Details of the convolutional layer is however, discussed separately in sub-section 2.2.5.

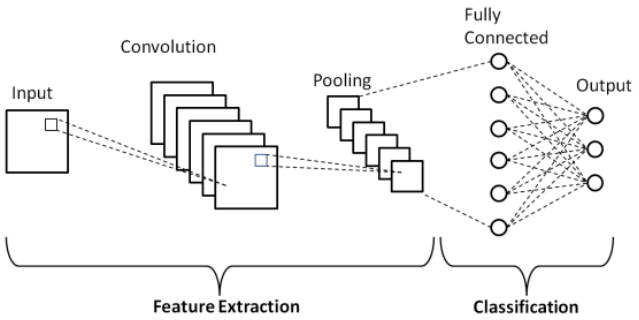


Figure 2.7: Basic CNN architecture [4]

The layers used to construct a CNN are briefly described below. As seen in the 2.7, the final layer a CNN is used for computing classification score, while the remaining network is used for feature extraction. It is to be kept in mind that the layers in a CNN have a volume rather than a size. A RGB image is represented as a 3D input volume and the CNN layers all transform the 3D volume to produce the final output. Many of the layers in CNN have parameters known as hyper-parameters, that determine and control its behaviour.

- **Fully Connected layer:** The fully connected or dense layer has already been defined in the context of a fully connected neural network. A dense layer is one where the neurons are fully connected to the activations of the previous layer. In a CNN, fully connected layers usually serves as the output layer and are responsible for calculating the classification score. The output size of such a layer is equal to the total number of classification class labels. The input to the fully connected layer are generally flattened and connected to all the neurons. The activations for this layer is computed by first computing the dot product between activations from previous layers and input data, and then, by the addition of the bias to the dot product output. Figure 2.1 depicts the computation of activation for a fully connected network.

- **Convolutional layer:** This has been discussed in details in 2.2.5.

- **Pooling layer:** This layer is generally added after a convolution layer. Pooling layers reduces the spatial size of its input volume. The principle behind this layer is to approximate the values of pixels in an area (given by filter size) by replacing all the pixels with the value of the highest activated pixel (in case of max pool). Figure 2.8 shows the max pool operation. This action reduces the number of computation required along with the chances of overfitting of the model. Popular pooling strategies include: max pool, global average pool, L2-norm pool. the pooling layer has two hyper-parameters, namely, filter size (or spatial extent) F and stride S. Values of both of these parameters are generally selected as 2. Given an input volume W_1 x H_1 x D_1, and considering a pooling layer using max pool strategy, the output volume (W_2 x H_2 x D_2) can be calculated as [27, Module 2]:

$$(i)W_2 = (W_1 - F)/S + 1; \quad (ii)H_2 = (H_1 - F)/S + 1; \quad (iii)D_2 = D_1$$

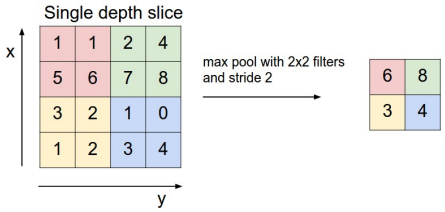


Figure 2.8: Max pooling operation [27]

- **Rectified Linear Unit (ReLU) layer:** This transform was proposed in the paper [34]. In a CNN, this forms the activation function applied after each convolution operation, which applies elementwise $\boldsymbol{max(0,x)}$ on the convolution output. The ReLU activation sets a threshold at 0 and selects only the values above 0. This operation does not alter the volume of the output from the convolution layer. It helps the network to train faster and solves the vanishing gradient problem[35]. The authors of [29] showed that ReLU performs much better that other transforms. A ReLU transform is illustrated by the figure 2.9.

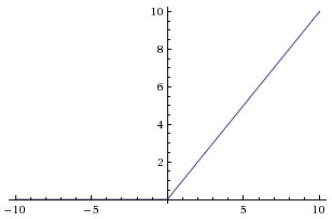


Figure 2.9: A Rectified Linear Unit transform [27]

- **Dropout layer:** This is a kind of regularization technique introduced by the paper [36]. The idea behind dropout layer is to randomly discard a set of neurons by setting them to

0. This prevents co-adaptation of features, which ensures that no single feature will have the power to influence the deduction.

- **Batch normalization layer:** The batch normalization technique was proposed in the paper [37]. Batch normalization layers are applied after fully connected layers and convolutional layers. This layer internally forces the input data to follow a Gaussian distribution. The advantage of this layer is that it helps in gradient propagation and use of higher learning rate while training.

2.2.5 Convolution Layer

This is the main transformation layer in a CNN. It can be imagined as a fully connected layer where (i) all neurons in the layers are not connected to all the activations from the previous layer, (ii) convolution operation is used instead of matrix multiplication to calculate neuron activations and (iii) convolution layer is composed of multiple filters or kernels of predefined volume, that represents the receptive field of the layer. This receptive field represents the section of input volume that interacts with the local neurons of the layer.

In fact, the neurons in a convolutional layer are sparsely connected and have a local connectivity. The initial convolutional layers learn about low level local features like edges and curves. Then, the intermediate layers learn to identify larger and more global patterns based on the lower layers. Based on this hierarchy, the final layer observes the global features that span much larger sections of the image.

Convolution Operation

Mathematically, a convolution operation is represented as: $\boldsymbol{s(t) = (x * w)(t)}$, where the $*$ stands for the convolution operation [25]. In case of a CNN, the x is the input data, w stands for the filter or kernel and the output is the feature map of a CNN. In a CNN, all of these parameters are multi-dimensional in nature. If the time t is considered a discrete distribution $(t(a))$, then it can be considered that the values of x and w correspond to particular intervals of time. Thus, in machine learning, convolution operation can be represented as:

$$s(t) = \sum_{a=-\infty}^{\infty} x(a)w(t-a) \tag{2.2}$$

Now, since the data in use is a 2D image (I), the kernel(K) will be 2D as well and thus the equation 2.2 can be revised for 2D images as given by equation 2.3 [25].

$$S(i,j) = (K * I)(i,j) = \sum_{m}\sum_{n} I(i-m, j-n)K(m,n) \tag{2.3}$$

The convolution layer comprises of filters or kernels that engage in the data transformation. The kernel has a specific volume, known as the receptive field. The movement of the kernel across the input can be described as a sliding window. The kernel thus, interacts with only specific parts of the input volume at a time, to produce the output volume. The figure 2.10 shows the interaction between the receptive fields and their weights, along with the specific areas of the

input volume along with the output volume. Thus the output volume can be calculated with the knowledge of the hyperparameters.

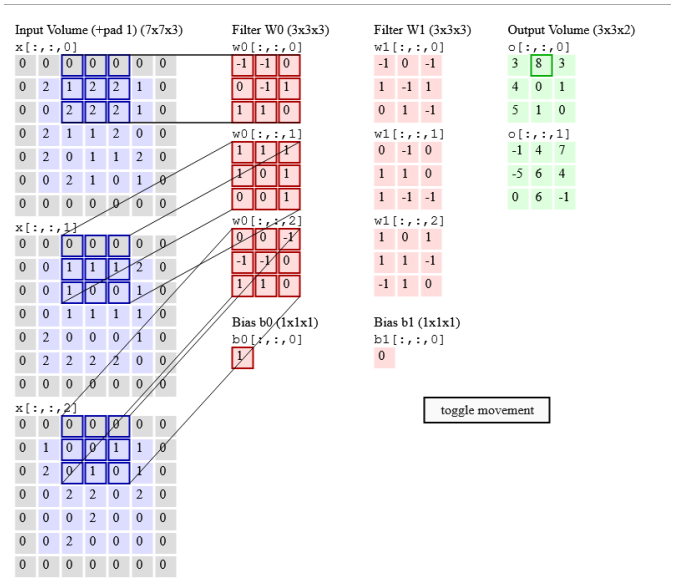


Figure 2.10: Example of a convolution operation [27]

Hyperparameters and output volume in convolution

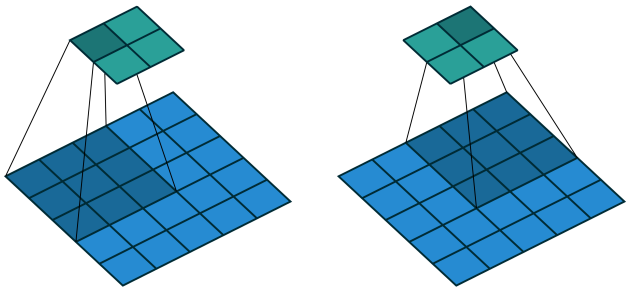
Convolution operation has a few hyperparameters that control or determine the behavior as well as the feature map of the layer. These are as below: like the kernel size, number of kernel, stride and padding.

- **Filter/kernel size or receptive field**: This gives the size or volume of the filter. It is smaller than the input volume and controls the number of localized neurons or pixels processed at a time. This parameter is also known as spatial extent of the filter. The depth of the filter volume is usually same as the input volume, i.e for a 3D input volume, there would be a 3D receptive field. In the figure 2.10, the filter size(F) are (3x3x3) [27].

- **Number of filters or kernels**: In a CNN, there are multiple kernels, each with a different set of weights, that process the input. The total number of kernels play an important role in determining the final output volume. In figure 2.10 number of kernels(K) is two [27].

- **Strides**: As the kernel interacts with the input volume, it has the option to (i) either select a set of adjacent pixels or neurons with the same area or volume as that of the kernel, or (ii) to skip a predefined number of pixels or neurons and then carryout the operation. The (ii) option is carried out using the property known as strides. The stride(S) in figure 2.10

is 2 [27]. Figure 2.11a shows the starting position of the convolution operation and figure 2.11b shows the operation after applying stride=2.



(a) Initial position (b) Applying stride=2

Figure 2.11: Convolution with stride [38]

- **Padding**: A normal convolution operation produces a smaller output volume than the input volume. This can reduce the input dimension very quickly and thus prevent creation of deep networks. Paddings are simply a layer of additional pixels (with value as zero) applied along the outside boundary of the input volume in order to preserve the volume. In figure 2.10 the padding (P) is 1 [27]. Figure 2.12 gives a visual representation of convolution with padding[38].

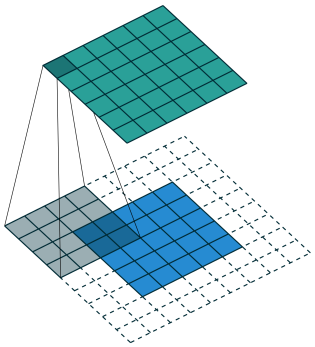


Figure 2.12: Convolution with padding [38]

- **Dilation**: Dilation is a hyper-parameter applied on the filter that makes it non-contiguous. This can help in capturing much more spatial features in lesser number of layers. Dilation does not influence the output volume. In figure 2.10, the filters are not dilated. [27]. The figure 2.13 shows a dilated convolution.

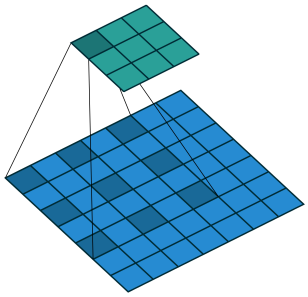


Figure 2.13: Convolution with dilation [38]

Depending on the value of one or more of these hyperparameters, the kind of convolution can vary. The size of the output can this be calculated using a formula: $(W - F + 2P)/S + 1$, where W is the input volume. Using this formula, the output volume of a CNN can be calculated. If W = 10, F = 3, S=1,P=0, the resultant volume is: 7.

2.2.6 Types of convolution

In general, when convolution operations or CNNs are mentioned, it stands for 2D convolution. However, there are few other variants and they are important in the context of this thesis. These variations are primarily based on how the convolution filter moves across the input volume,i.e. across the number of axes of movement of the kernel. It also depends on the type (image, time series) and nature of the data. Some of the different types of convolutions essential for this thesis, have been enumerated follows:

- **2D Convolution:** This is the most general case of convolution and most examples of convolution, unless otherwise specified, belong to this category. This is used primarily for image data, which is represented in 3 dimensions, i.e. width, height, depth. In 2D convolution, filter sizes are also 2 dimensions. This signifies that the convolution operation is carried out across the width and height dimensions, while the depth is of the same magnitude as the number of channels in the image.

- **1D Convolution:** 1D convolutions, as the name suggests, operates along a single spatial dimension. It can be imagined as a simplified version of 2D convolution, with only one axis. The output data size is also 1 dimensional. It is best suited for time series data, text processing etc.

- **3D Convolution:** In this kind of convolution, the filter size is defined in 3 dimensions, i.e. it has additional depth component. This means that the kernel moves across the input in both the spatial and temporal axis. This produces an output volume of 3 dimensions. This is essential for detecting both spatial and temporal features and is used for low level feature extraction from 3D medical images, videos and for motion detection. Figure 2.14 depicts the 3D convolution.

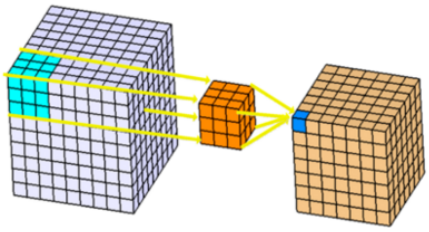


Figure 2.14: 3D convolution operation [39]

- **1x1 convolution:** 1x1 convolution was proposed in the paper [40] and used in the paper Inception network [31]. In this convolution, the filter shape is 1 x 1 x D, where D is the depth of the data as well as kernel. Considering input shape of W x H x D, and using N number of 1x1 filters, the output volume can be calculated as: W x H x N. 1x1 convolution helps to reduce the dimensions, especially along the depth axis, and thus, reduces the amount of computations needed [41]. The figure 2.15 shows the 1x1 convolution operation.

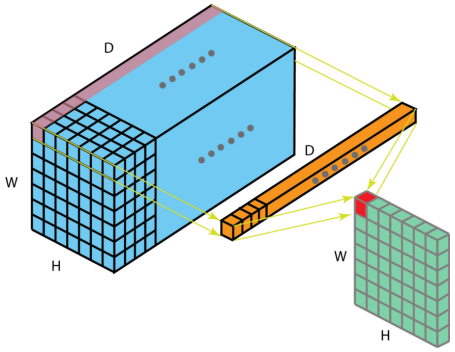


Figure 2.15: 1x1 convolution operation [41]

- **Dilated convolution or atrous convolution:** Dilated convolution is performed using a dilated kernel as described in sub section 2.2.5. This idea was introduced in the paper [42].This kind of filter effectively increases the receptive field with fewer computations, while keeping the output volume unaffected. The figure 2.13 shows how the dilated kernel moves across the input and the 2.16 shows the dilated receptive fields with different dilation values.

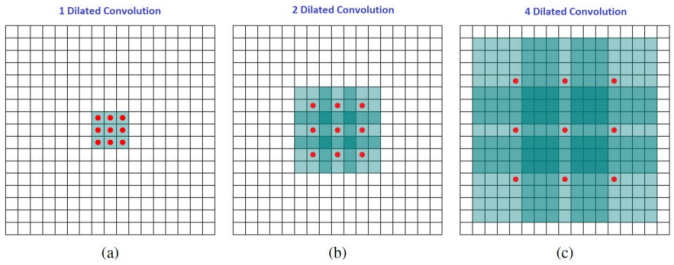


Figure 2.16: Receptive field with 1,2,4 dilations [41]

Apart from the above mentioned convolutions, there are a few more convolution types, for example, transposed convolution, depth-wise separable convolution, spatially separable convolution, flattened convolution, grouped convolution. However, for maintaining the focus and compactness of this thesis, these additional types have not been discussed here.

2.2.7 Visualizing CNNs: Intermediate Activation and Filters

Although the output volume of a convolution operation can be easily calculated, the actual low-level features learned at each step of the transformation and its influence on the final prediction, remains unknown. A part of this problem can however be addressed by visualizing the intermediate layer-wise activations and the filter learned in the process[27].

1. Visualizing intermediate activation means displaying the feature maps for each filter at each convolution and pooling layer that are generated during the forward pass of the training process. Each of the filters learn specific features, which are captured in these feature maps, and this provides a distinct view into the evolution of features emphasized and learned by every filter in each layer across the hierarchy of the network. [26].

2. Visualizing filters of convolution and dense layers means visualizing the weights learned by these filters. This gives an intuitive way to understand the patterns each filter has learnt to detect in an image. These features and shapes becomes more and more abstract as the depth of the network increases. The visualization of the filters is achieved by providing a blank input image to the network and performing gradient ascent on it. Next, the responses of each filter are maximized to detect the shapes to which the filters responded [26].

Out of these techniques, in this thesis, intermediate activation visualization technique has been used to evaluate the results of the experiments. More complex tools have been developed to visualize the internal representation of a CNN and detect the discriminative objects that contribute towards the prediction results. These tools have been discussed in the section 2.3

2.2.8 Application of Machine learning in remote sensing images

In the previous sections details of various convolution types and hyperparameters have been presented. They are some of the fundamental components of a machine learning algorithm and specific combinations of those are used depending on the scenarios and the kind of data. This

section presents an insight into several research papers and journals to highlight the different ways in which different classification tasks have benefited from the use of different types of convolution (other than 2D Convolution) and its hyperparameters. The discussion is carried out with respect to remote sensing domain, but a few examples from other domains have been also presented.

One of the major uses of remote sensing is in land use/cover analysis. In the paper [43], the authors have used data from Sentinel-2 satellite system to study the impact of its data on land use/cover classification. The authors inferred that use of support vector machine (SVM) and Random forest (RF) improved the classification result as compared to traditional methods. In the paper [44], the authors have analysed the effectiveness of transfer learning in remote sensing scene classification problems and concluded that transfer learning provides very accurate results.

In the paper [45], the authors have used a CNN to classify species of trees in a conifer forest from airborne HSI data. The model architecture was composed of a set of 3×3 convolutional layers followed by a final 1×1 convolutional layer for the output. Using the pixel-based classifier, the authors were able to identify 713 individual trees and achieved an improved f-score (0.87) compared to a model trained on natural images (f-score 0.64). The authors found that this model performed significantly better in identifying tree species, change of tree distribution and dead trees and pathogens. The authors of the paper [46], developed a multilabel classification model for deep learning using data augmentation technique to overcome the paucity of training data. This model showed an improved multilabel accuracy of 8% and f-score of 6.65% compared to standard models.

In the paper [47], authors Ji et al. implemented 3D convolution in a CNN to classify crops using multi-spectral multi-temporal images. They designed a 3D kernel that fits the nature of the data and then trained the 3D CNN to identify spatio-tempral characteristics of crops from the images. They argue that, although use of 3D CNN is not popular in machine learning, but due to the presence of temporal information in the data, it is much more suited than 2D CNN for this type of data. The temporal information provides information on the growth cycle of the crops. They organize the spectral bands into 3D tensors by stacking them along temporal dimensions. These 3D tensors are then used as input to the 3D CNN. They chose a 5 layer network, similar to VGGnet, where the 2D convolutions were replaced by 3D convolution. A similar model using 2D convolution was trained as well which was outperformed by the 3D CNN. The 3D CNN was more sensitive towards subtle discriminative features among crop types. A similar experiment was conducted in the study [48] using Landsat and Sentinel data. The authors compared two models (i) uses 1D convolution is spectral domain, (ii) uses 2D convolution in spatial domain. The 2D CNN provided better accuracy along with better class discrimination capability in identifying crop types.

In the paper [49], the authors have suggested a 3D CNN for human action recognition from video data. Due to 3D convolution, the model can extract spatial and temporal information concerning the motion across multiple frames. The novel 3D CNN proposed in this paper, generates several channels of data from the video and then applies convolution and pooling on each independent channel. In the final layer, the feature map from all channels are combined to produce a 128-D feature vector (for 128 actions) and the a linear classifier is applied on it. This model performed satisfactorily without any need for feature identifier.

In the paper[50], a 3D CNN was demonstrated to surpass a fully connected deep neural network (FcDNN) in classification performance and feature extraction while classifying fMRI volume data. The authors designed a model with the LeNet-5 [28] architecture and replaced the 2D convolutions with 3D convolution to perform their experiments. Their model showed a distinct reduction of error rate, slight improved performance and exceptional feature extraction capability compared to the FcDNN model.

In the paper [51], the authors proposed a novel architecture to identify and segment small objects out of high dimensional data using dilation. Higher dimensional images comprises of many small objects located close together. This poses a challenge in effective segmentation. This has been addressed by the authors. Thus, a new model for local feature extraction (LFE) has been proposed. Dilation expands the receptive field without loss of resolution. They divide their architecture into three modules, out of which: (i) to extract global features, thus using gradually increasing dilation, (ii) to extract local features, there by gradually reducing dilations. The proposed model was able to identify small objects in multiple databases with high accuracy.

In the paper [52], the authors have proposed a new model called CSRNet for highly congested scene scene recognition. The underlying motivation of the authors can be described as: " capturing high-level features with larger receptive fields and generating high-quality density maps without brutally expanding network complexity"[52]. In the paper, the proposed network has two parts, a 2D CNN feature extractor and a dilated CNN to create large receptive fields. The authors concluded that the model was successful at creating high-quality feature maps to identify crowded scenes and performed better than state of the art models for the same tasks.

Dilated convolution is most popular for image segmentation tasks. However, the authors of the papers [10] and [53] have propounded the use of dilation in hyperspectral image classification. Cui et al. [10], proposed (i) a light-weight block using multiple residual connections and dilated convolution for feature extraction and (ii) a CNN for feature extraction using multiple receptive fields, and multi-scale spatial features. In [53] the authors proposed to create a novel CNN using transposed and dilated Convolutions. Although highly contrasting techniques were used, both the image classification models were found to be very effective for HSI classification.

These studies, though not an exhaustive list, provide an idea into the kind of scenarios that might be appropriate when looking for specific data features. For this thesis, there are a number of factors to consider while choosing appropriate convolution type or values for convolution hyperparameters. However, based on the research papers cited in this section, this thesis proposes to study the effect 3D convolution and dilation on multispectral multi-label data.

2.3 Explainable AI

Machine learning has made significant progress and has achieved very high level of competency in solving complex problems over the last few years. For some specific tasks, it has started rival with humans in its problem-solving capabilities.

As seen in the paper [54], the authors reached 99.15% accuracy in understanding traffic symbols using a CNN. In [55], the authors proposed a model that achieved 98.52% accuracy in the Labelled Faces in the Wild (LFW) benchmark, thereby exceeding human level performance.

Machine learning has also enabled computers to understand strategies behind complex games

and play games as well. In the paper [56], the authors have presented a novel model called deep Q-network, that uses reinforcement learning to discern policies directly from sensory input. This deep Q-network was able to play the Atari 2600 games. In 2016, Silver et al. [57] presented Alpha Go, which was learnt the rules of Go through deep neural networks and reinforcement learning and defeated the human champion. But Alpha Go was defeated by the advanced model Alpha Go Zero [58] which had learnt the rules of the game by pure reinforcement learning and without any human supervision, or knowledge of any human expert moves.

An impressive example is Wavenet [59] which is a deep neural network that is able to generate speech and is highly comprehensible when used for read text-to-speech generation.

But despite these incredible advancements, the trust in machine learning system remains quite low. The problem is due to the fact that, as the performances of machine learning systems have improved, they have increasingly become more and more complex and their internal interpretations of the data have become more opaque to humans. As a result, machine learning systems are often called "black boxes".

Explainable AI (XAI) has emerged from this attempt to understand the decision making process of a ML tool or algorithm and obtain explanations for its decisions. It enables us to assess the results for fairness, quality, safety, and also to ensure we can learn previously undiscovered patterns and characteristics in our data. It can also help in correcting any inadvertent mistakes introduced by the algorithm or by data. In this way XAI can help in establishing trust in AI based systems. The paper [60] discusses the need for such processes in medicine.

2.3.1 Need for interpretability in ML

While the latest advancement in machine learning algorithms and tools are the reason for progress in this domain, the data used in the training process is equally important. Data can introduce mistakes, spurious correlations and biases in the network. When a deep learning model trains on such data, it imbibes these biases thus leading to compromised or dubious results. This makes human verifiable results a necessary requirement. In this section, a few examples of such spurious correlations and mistakes have been presented.

In the article [61], the authors trained a deep neural network using a large number of images and then attempted to visualize the actual features learnt from these images. This process revealed that the model learned to recognize a dumbbell along with an arm as shown by the figure 2.17. Thus, such a model will always associate the presence of an arm with the object dumbbell and perhaps will ignore instance of the dumbbell when it does not appear held by an arm.

Figure 2.17: Figure showing model learnt to recognize dumbbell along with an arm [61]

In the article [62], the authors trained a classifier using images of wolf with snow in the background and husky dog without any snow in the background, in an attempt to investigate the features that contribute to the classification. It was revealed in the study that the model had learned to associate the presence of snow with the class wolf and its absence as class husky dog. Figure 2.18 show the result from the paper.

(a) Husky classified as wolf (b) Explanation

Figure 2.18: Figure showing model learnt to associate wolf with snow [62]

2.3.2 Explainability and Interpretability in Machine Learning

Though the concept of explainable AI has been introduced and its necessity elucidated, it is important to clearly define the term interpretability. The term interpretability in ML was defined in the paper [63] as: "In the context of ML systems, we define interpretability as the ability to explain or to present in understandable terms to a human." [63]. Lipton et al. in [64] provided a comprehensive explanation of interpretability in supervised learning and defined properties of an interpretable model. According to Lipton et al., one of the desired properties of an interpretable system is Post-hoc Interpretability.

According to Montavon et al. in [65], post-hoc interpretability is " a trained model is given and our goal is to understand what the model predicts (e.g. categories) in terms what is readily interpretable (e.g. the input variables)"[65]. Montavon et al. also provided definitions for interpretation. Interpretation has been defined as "An interpretation is the mapping of an abstract concept (e.g. a predicted class) into a domain that the human can make sense of."[65]. Explanation has been defined as "An explanation is the collection of features of the interpretable domain, that have contributed for a given example to produce a decision (e.g. classification or regression)."[65].

2.3.3 State-of-the-art techniques in Explainable AI

This section describes few of the state-of-the-art explainability methods. Here, a highlight of each method has been presented using their underlying motivation, proposed formulae, and example images.

1. Saliency Map Analysis

Simonyan et al. [30] proposed on creating a Saliency map based on gradient of input image to visualize a CNN. Two different visualization techniques has been proposed by the authors of this paper. First one is for visualizing learned features by deep convolution models and the second technique is for generating class object localization in an image using Saliency maps.

For the first technique, named as "Class Model Visualization"[30] in the paper, the approach is to optimize the input image. The motivation behind this is to try to **maximize the class score** learned by the CNN model. The authors have obtained the maximum classification score $S_c(I)$ for input image I and class c. This score was then used to generate a representative image of the class features learnt by the CNN during training. The optimized class score is formulated as

$$argmax_I S_c(I) - \lambda ||I||_2^2$$

where λ is the regularization parameter. Results generated using this technique has been presented in figure 2.19.

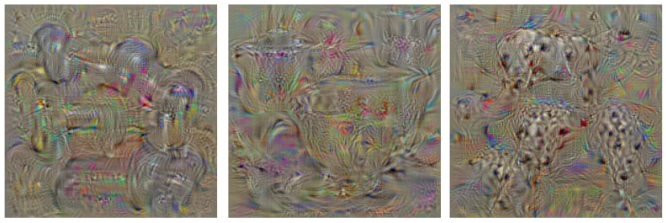


Figure 2.19: Representations of a dumbbell, cup and dalmatian dog [30]

For the second technique, Simonyan et al. suggested one pass of **back-propagation** to be run on the trained CNN to create the class-specific saliency map for the input image. The authors explained that these kinds of saliency maps help in localization of objects in weakly supervised networks. This technique relies on ranking the pixels of image to understand the contribution of each pixel towards the prediction. The authors performed a first-order Taylor series expansion of the classification score $S_c(I)$ for the class c, specific to the image I_0, which is given as:

$$S_c(I) \approx w^T I + b_c$$

Here w is the derivative of score S_c for image I at point I_0 and is given by

$$w = \left.\frac{\partial S_c}{\partial I}\right|_{I_0}$$

The class score derivative also represents the pixels that are most sensitive or discriminative towards class change and thus it is highly representative and localizes the class object correctly in the image. The saliency map was computed from the derivative w for a grey-scale image, where w has m rows and n columns. For a natural or RGB image, w has an additional channel

component c. The basic principle for the map $M_(i,j)$ for grey-scale and natural images are respectively:

$$M_{(i,j)} = |w_{h(i,j)}| \quad \text{and} \quad M_{(i,j)} = max_c|w_{h(i,j,c)}|$$

The figure 2.20 shows the saliency maps generated for the input images using CNN. [30]. They show the areas related to the object car, in the first image, and fish, in the second image, are highlighted in the respective saliency maps created from the images.

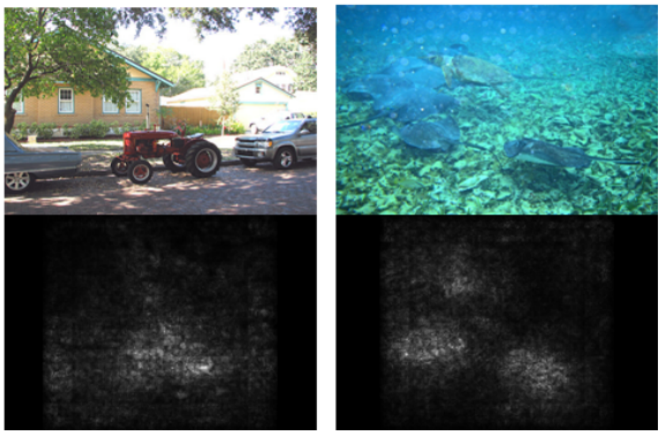


Figure 2.20: Saliency Map generated for images. [30]

2. Deconvolution

In the paper [15] authors Zeiler et al. tried to understand the superb performance of deep convolutional neural networks in the ImageNet [66] classification challenge. To this end, they propose to use deconvolution [67] to visualize the intermediate activations and structures in an input image that influence feature maps. The strategy behind deconvolution is to map the activations of convolution layers back to the pixel space and visualize the input patterns that influenced the activated neurons. For the experiment, the authors attached deconvolution layers with every convolution layer. Since deconvolution can be imaged as the opposite of convolution operation, the process also involved unpooling, rectifying of the activation values and application of filter to generate the features. Once the network is trained, the data flow is then reversed and the features are generated by passing through the transformations given in the figure 2.21.

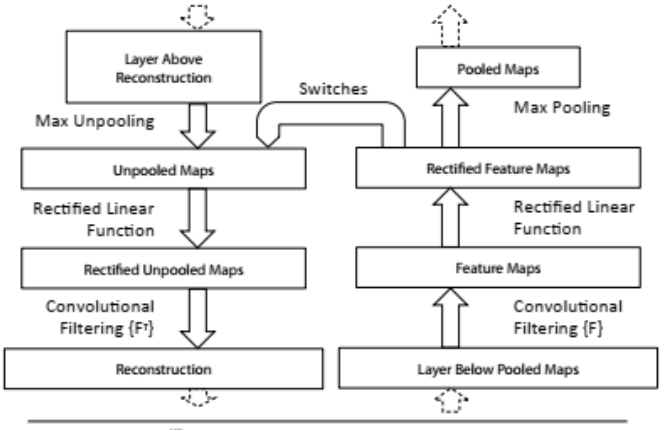
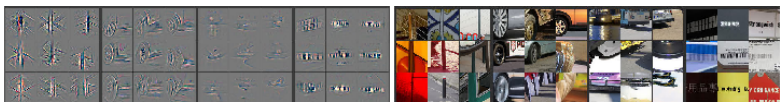
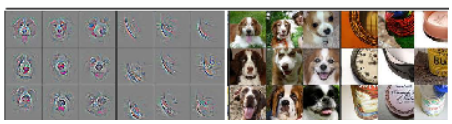


Figure 2.21: The step-by-step operations involved in Deconvolution[15]

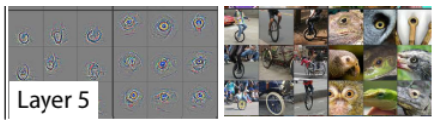
The results showed that this process was able to decipher at each level the properties of the image that were deemed most noteworthy by the network and caused activation of the feature map. A stark contrast in the nature of those structures based on the level of convolution layer could also be observed. A small example of the results from [15] has been presented in figure 2.22



(a) Layer 3 visualization



(b) Layer 4 visualization



(c) Layer 5 visualization

Figure 2.22: Layer-wise Deconvolution results [15]

The images show the various characteristics that were identified at each layer. In figure 2.22a, different shapes related to wheels, signposts etc were identified by the process in the layer 3 of the model. In the layer 4 2.22b, the model can identify a mouth of a dog very successfully. In the layer five 2.22c, the model identifies very specific shapes like the different postures of a bicycle wheel and the eyes of different bird.

3. Guided Backpropagation

The authors of the paper [68] came up with a new technique. called guided backpropagation, for visualizing the features learnt by a CNN, from an attempt to evaluate the efficacy of the order in which the layers in most deep CNNs are usually arranged. In this paper Springenberg et al.

investigated the impact of max pooling layer and its role in improving classification performance. The proposed visualization approach is a variation of the deconvolution [15] approach discussed in 2.3.3. The authors demonstrated that a max pooling layer can be replaced by a convolutional layer with stride=2. They tested their theory by training their CNN, with convolution layer with stride=2, with CIFAR-10, CIFAR-100 [69] and ILSVRC-2012 ImageNet [70] databases. As a method of evaluation, they modified deconvolution, since it requires max pooling layer, and replaced it with a technique that combines backpropagation and deconvolution. This method prevents backpropagation of negative gradients for neurons, that they wished to visualize. This is because the negative gradients of those neurons could reduce the intensity of the activations of the chosen neurons in the higher layer. This could have an adverse effect on the visualization. The authors compared backpropagation, deconvolution and guided backpropagation using two models, (i) one was trained without max pooling layer, but max pool layer was inserted in place of the strided convolution layer before visualization; (ii) the second one did not use any pooling in training or visualization. The results are given in figure 2.23

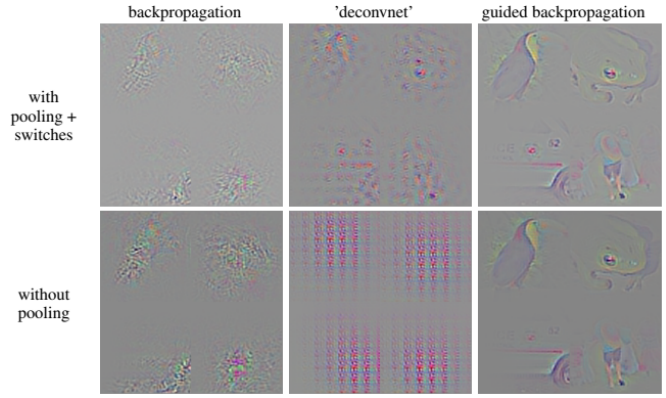


Figure 2.23: Guided Backpropagation vs. deconvolution and backpropagation [68]

From the figures 2.23, the effects of backpropagation, deconvolution and guided backpropagation can be compared. Although backpropagation highlighted sections of the image, the results were not distinct and could not be recognized as an object. In case of deconvolution with pooling and switches, the results were better and more comprehensible. However, deconvolution without pooling failed to generate any identifiable feature. The results of guided backpropagation using both of the techniques, were the most distinct. Object classes that were deemed as important could be easily discerned and identified. This shows the superior performance of the guided backpropagation comapred to the other two techniques.

4. CAM, Grad-CAM, Guided Grad-CAM

CAM or Class Activation Maps is a visualization technique proposed by Zhou et al. in the paper [71]. This is based on the concept of using Global Average Pool (GAP) to improve localization capability of a CNN. The authors propose a technique which, along with the use of global aver-

age pooling, enables the network to localize the discriminative areas in an image that contributed to the prediction. This is achieved in a single forward pass without retraining the model. The authors describe class activation map as "class activation map for a particular category indicates the discriminative image regions used by the CNN to identify that category"[71]. In their model architecture the authors precede the final dense layer with a GAP. The CAM is created by projecting the learned weights of the final layer on the convolution feature maps. The mathematical deduction is given below [71]:

- (i) For the location (x,y), the activation of the final convolution layer is given by $f_k(x;y)$.
- (ii) After applying GAP to the above, the output for corresponding unit is: $F_k = \sum_{x,y} f_k(x,y)$.
- (iii) Thus the softmax input for class c, $S_c = \sum_k w_k^c F_k$, w_k^c gives the importance of c for the unit k.
- (iv) The output of softmax for class c, is $P_c = \frac{\exp(S_c)}{\sum_c \exp(S_c)}$.
- (v) The authors project back the weights of the output layer on the feature maps from last convolution layer to obtain the CAM. This is achieved by plugging equation (ii) to S_c. Thus, S_c is given by:

$$S_c = \sum_k w_k^c F_k = \sum_{x,y} \sum_k w_k^c f_k(x,y) \tag{2.4}$$

- (vi) Thus the CAM for area (x,y) is given by:

$$M_c(x,y) = \sum_k w_k^c f_k(x,y) \tag{2.5}$$

The resulting weighted average for each unit represents the area of highest discriminative value for the chosen class. The result of this process has been depicted in the figure 2.24. It shows the highest discriminative sections (highlighted with red) of the images that has been identified by CAM. It can observed from the images that although the same object was detected in many of them, CAM correctly identifies the exact location of the objects in the image, thus proving its superb localization capability.

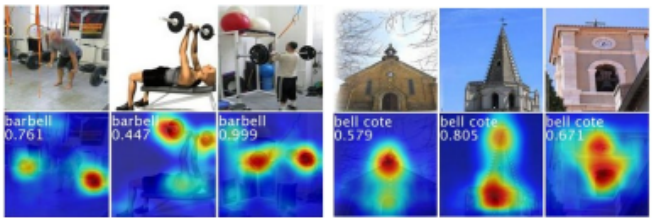


Figure 2.24: Class activation maps with most discriminative regions highlighted in red [71]

The Gradient-weighted Class Activation Mapping (Grad-CAM) technique was proposed in the paper [14] and is based on CAM [71]. The authors of [14] proposes an improvement over

CAM [71] by removing the constrain of using global average pooling layer (convolution feature maps $\rightarrow$ global average pooling $\rightarrow$ softmax layer) and thus, they makes it applicable to a wider range of architectures. The authors Selvaraju et al. define Grad-CAM as a method to calculate importance of every neurons in the last convolution layer with respect to a class prediction based on the gradients value of the last convolution layer. They use it to generate a localization map that identifies the discriminative regions. They further combine Grad-CAM with pixel-space gradient visualization techniques to produce a new visualization technique called Guided Grad-CAM. For Grad-CAM, neuron importance α_k^c is given by:

$$\alpha_k^c = \frac{1}{Z}\sum_i\sum_j \frac{\delta y^c}{\delta A_{i,j}^k} \tag{2.6}$$

While calculating α_k^c and back-propagating with respect to the activations, a series of matrix multiplications were performed. A weighted combination of activation maps was obtained as a result and ReLU activation is applied on it. This produces the class-discriminative localization map Grad-CAM $L_{Grad-CAM}^c$. This is depicted by [14]:

$$L_{Grad-CAM}^c = ReLU(\sum_k \alpha_k^c A^k) \tag{2.7}$$

This generates a heatmap with the same size as the feature map comprising of the highly discriminative area of the image. This can be seen in figure 2.25. It shows the original image 2.25a, along with images containing heatmap indicating classes dog 2.25b and cat 2.25c respectively. This proves the superb object detection and localization capability of Grad-CAM.

(a) Original image (b) Heatmap for class 'dog (c) Heatmap for class 'cat'

Figure 2.25: Grad-CAM results depicting individual classes [14]

5. Layer-wise relevance propagation (LRP)

The concept of layer-wise relevance propagation was presented in the paper [13]. It is based on Taylor decomposition, proposed in [72]. This goal of this paper was to enable users to learn the contribution of every single pixel towards the prediction for the image. The method proposed method follows a set of propagation rule and propagating backwards the relevance score of the prediction from the last layer towards the input layer. The authors have defined a Relevance score (R_d^{l+1}) for dimension z_d^{l+1} of each vector z at layer $l+1$. For two neurons j and k in

two consecutive layers, and $z_{(j,k)}$ as the influence of neuron j on neuron k,the propagation of relevance can be expressed as:

$$R_j = \sum_k \frac{z_{(j,k)}}{\sum_j z_{(j,k)}} R_k \tag{2.8}$$

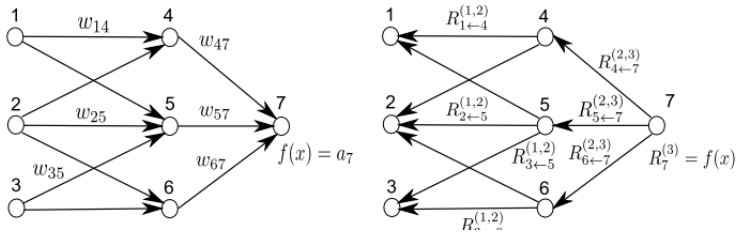


Figure 2.26: Flow of relevance from upper to lower layers [13]

The figure 2.26 shows how the relevance propagation takes place from one layer to another in both forward and backward direction. The relevance of each neuron in the backward propagation is considered while building the rules governing LRP. From the flow of relevance in figure 2.26 can be generalized as:

$$R_k^{(l+1)} = \sum_{\substack{i:\ i\ is \\ input\ for \\ neuron\ k}} R_{i\leftarrow k}^{(l,l+1)} \tag{2.9}$$

One essential rule of LRP is that equation 2.9 must be preserved. Now, since neuron activation functions are non-linear, using the pre-activation $z_{i,j}$, relevance decomposition can be expressed as a ration of local and global pre-activations. This is given by:

$$R_{i\leftarrow\ j}^{l,(l+1)} = \frac{z_{(i,j)}}{z_j} R_j^{(l+1)} \tag{2.10}$$

Using further deductions that are not discussed for the sake of brevity in this thesis, the authors deduced relevance propagation :

$$R_{i\leftarrow\ j}^{l,(l+1)} = R_j^{(l+1)} \cdot (\alpha \cdot \frac{z_{i,j}^+}{z_j^+} + \beta \cdot \frac{z_{i,j}^-}{z_j^-}) \tag{2.11}$$

The paper [73], enumerates a few different types of LRP all created on the equation 2.11. These are: (i) Basic Rule (LRP-0), given by

$$R_j = \sum_k \frac{a_j w_{(j,k)}}{\sum_{0,j} a_j w_{(j,k)}} R_k \tag{2.12}$$

(ii) Epsilon Rule (LRP-ε), given by

$$R_j = \sum_k \frac{a_j w_{(j,k)}}{\varepsilon + \sum_{0,j} a_j w_{(j,k)}} R_k \tag{2.13}$$

and (iii) Gamma Rule (LRP-γ), given by

$$R_j = \sum_k \frac{a_j(w_{(j,k)} + \dot{\gamma} w^+_{(j,k)})}{\sum_{0,j} a_j(w_{(j,k)} + \dot{\gamma} w^+_{(j,k)})} R_k \tag{2.14}$$

The paper [73] also presented the generic rule, given by,

$$R_j = \sum_k \frac{a_j \rho(w_{(j,k)})}{\varepsilon + \sum_{0,j} a_j \rho(w_{(j,k)})} R_k \tag{2.15}$$

In this paper, the authors also provided some guidelines on which rule is most suited for which layer of a CNN depending on the depth of the layer. This has been shown in the figure 2.27. It shows the different characteristics of the image identified by the different LRP rules. A visualization library, called iNNvestigate, [74] was developed to implement the different LRP rules. This library has also been used in this thesis. It must be mentioned here that a way to apply different LRP rules at different rules by using any software library did not exist at the time of writing this thesis.

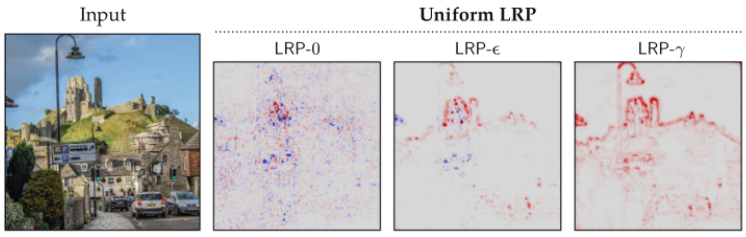


Figure 2.27: Result of different LRP rules [73]

2.3.4 Application of explainable AI in remote sensing images

Explainiability is a critical factor in generating trust on machine learning applications across domains. This section discusses a number of recent scientific papers that have used XAI techniques to evaluate complex classification problems. For the purpose of this thesis, Grad-CAM [section 2.3.3], LRP [section 2.3.3] and visualization of intermediate activations [section 2.2.7] have been used to analyse the experimental results. Thus studies based on these methods have been presented below.

In the paper [75], the authors have proposed a deep learning based multi-label classification model that can detect multiple types of lesions in diabetic retinopathy (DR) fundus images, and used Grad-CAM to identify different locations of different kinds of lesions. The underlying architecture was based on ResNet [32]. They were able to achieve sensitivity of 93.9% and specificity of 94.4% and were able to pinpoint the locations of lesions.

In the article [3] have used Grad-CAM to visualize the classification output of a ResNet based CNN that uses a special attention mechanism for improved feature extraction.

In the paper [76], the authors have used LRP technique to explain the decisions of the deep neural network used for MRI-based Alzheimer's classification. They compared the performance

of guided backpropagation and LRP and concluded that for Alzheimer's disease classification using MRI data, LRP was able to successfully highlight the positive contributions.

Phung et al in the paper [4] visualized filters and feature maps to evaluate the performance of high accuracy of classification model based on cloud patches.

In the paper [77] have recommended the use of LRP as a tool to interpret deep learning models used for geoscience research. They used LRP to analyse the performance of a sea aurface temperature based deep learning model in interpreting climate patterns like El Niño and La Niña. They also studied the efficacy of LRP in identifying patterns in a model trained to predict sea surface temperature. in and to study seasonal ocean patterns. In both the cases, LRP provided excellent performance in interpreting the deep learning models.

Based on this discussion on the different state-of-the-art visualization techniques, for the purpose of this thesis (i) LRP, (ii) Grad-CAM have been chosen. Further information regarding the exact methods used and experimental setup, are discussed in section 4.

3 Proposed Work

This chapter presents the proposed work for the thesis. This thesis is an exploratory study of the effects of convolutional layer. The goal is to understand how the use of different convolutional parameters, changes the internal representation of the images and consequently, impacts the overall classification result.

In the chapter 2, particulars of different convolutions and its parameters have been discussed. Several scientific papers implementing different convolutions have been presented in support of their efficacy. Based on these results, a number of ideas have been proposed to attain the goal of this thesis.

The ideas that this thesis proposes to explore are as follows:

- As a very first step, this thesis wants to establish a baseline for classification performance with respect to the data used and the objective of this study. For this purpose, a model using 2D convolution will be used to classify the data. This is because, 2D convolutions are the conventional choice for remote sensing image classification problems. This reference will help in comparing subsequent experimental models and evaluating their performances.

- In some of the works discussed in 2.2.8, it has been shown that 3D convolution surpassed the performance of 2D convolution-based models. As demonstrated in the papers [50] and [47], 3D convolution provided superior results compared to comparable network architectures. Although, the data used in this thesis does not have temporal information, however this thesis makes an assumption that 3D convolution should be able to use the combined information from multiple spectral bands to extract more detailed information on the land/cover. This will also improve the classification performance compared to the 2D convolution reference model. Hence, this study proposes to test and compare the efficacy of 3D convolution with respect to 2D convolution in remote sensing image classification.

- As discussed in the 2.2.8, dilations are generally used for image segmentation tasks [51] or for interpreting crowded scenes [52]. However, remote sensing images also contains a mix of very fine-grained details of land-forms belonging to multiple classes. Thus, it would be interesting to examine the effect of dilation on the data. It is to be noted that the data has multiple resolutions. So, dilation may not be suitable for the data coming from the lower resolution satellite bands. Thus, this study wants to examine if dilation has any impact on classification result for such images. To this effect, multiple CNNs with dilated convolutional layers, at different levels and branches, will be created. The results from these models will be compared and analysed.

- In order to achieve the two tasks described above, this thesis proposes to visualize the features learnt the classifiers by visualizing the intermediate activations and by using LRP, Grad-CAM techniques.

4 Dataset Description and Experimental Setup

In this chapter, details of the experimental setup and dataset used for this thesis have been discussed. The experiments are designed according to the plan of action presented in chapter 3. This chapter is divided into four sections: 4.1 describes the data used in this thesis and briefly discusses the of train/test split for classification; 4.2 presents the baseline architecture, and the modifications done for creating the experimental models using dilation and 3D convolution; 4.3 discusses the different parameters used and hardware and software requirements for this thesis; 4.4 discusses the metrics used for evaluation of results.

4.1 Dataset

For this thesis, the BigEarthNet [1] dataset has been used. The BigEarthNet is an archive of multispectral image patches from the Sentinel-2 satellite system. These image patches are multi-labelled using land-cover labels from the CORINE land cover map of 2018. The image patches have been annotated with a total of 19 class labels. The BigEarthNet archive comprises of 590,326 image patches with three spatial resolutions, namely (i) 10m bands representing 120 x 120 pixel area, (ii) 20m bands representing 60 x 60 pixel area and (iii) 60m bands representing 20 x 20 pixel area. The patches also comprises of total 13 bands which includes, four 10m bands, six 20m bands and two 60m bands. The table 2.1 provides information on the bands and their individual spatial resolutions. [1] provides the list of class labels and number of image patches contained in each. The same information has been provided in the table 4.1.

For the purpose of this thesis, the archive of 590,326 images has been divided into training, validation and test sets. Preprocessing of the data was not required as the data already excluded the images fully or partially covered with snow, cloud and cloud shadow. The distribution of classes per set provided in [1] has been used in this thesis as well. Thus, the training, validation, test sets have 269,695, 123,723 and 125,866 images respectively. The distribution of classes in these individual datasets is available in [1] for reference. These three image datasets were converted into three tfrecord files which were used as input files during the training, validation and test stages of classification.

4.2 Network Architecture

As discussed in the chapter 3, this thesis plans to analyse the efficacy of 2D and 3D convolutions, and dilation parameter in classification of multi-spectral, multi-labelled land-cover images. To

Class Names	Number of images
Urban fabric	74,891
Industrial or commercial units	11,865
Arable land	194,148
Pastures	98,997
Permanent crops	29,350
Complex cultivation patterns	104,203
Land principally occupied by agriculture, with significant areas of natural vegetation	130,637
Agro-forestry areas	30,649
Broad-leaved forest	141,300
Coniferous forest	164,775
Mixed forest	176,567
Moors, heathland and sclerophyllous vegetation	16,267
Transitional woodland-shrub	148,950
Beaches, dunes, sands	1,536
Natural grassland and sparsely vegetated areas	12,022
Inland wetlands	22,100
Coastal wetlands	1,566
Inland waters	67,277
Marine waters	74,877

Table 4.1: BigEarthNet class labels with label count [1]

this end, at first a baseline model has been designed and based on this model several experiments have been designed with a combination of 2D and 3D convolutions and dilation.

4.2.1 Baseline architecture

As the initial step, a baseline architecture is defined for this thesis. All the other experiments are designed with respect to this architecture. This model is then used to classify the multi-label data and that result is considered the baseline for all other results in this study.

In this thesis, an alias has been assigned to each experimental model to highlight the architectural change made in it with respect to the baseline model. Thus, the alias for the baseline model has been chosen as '2DConv_base' model.

The baseline architecture used in this thesis is inspired from the K-branch architecture introduced in [2]. The detailed architecture of K-branch model was presented in [1]. For the purpose of this study the architecture has been adapted according to the data at hand.

The 2DConv_base model is a CNN with three branches, one for each of the three spatial resolutions of the data. These branches are named as, 10m branch, 20m branch and 60m branch respectively, to indicate the spatial resolution of the input data. The input data for each branch has the dimension as: (i) 120x120, (ii) 60x60, (iii) 20x20. The schematic block diagram of the

whole 2DConv_base model used in this thesis is provided in the figure 4.1. The actual detailed architecture can be found in the appendix section A.1. The particulars of each branch are as follows:

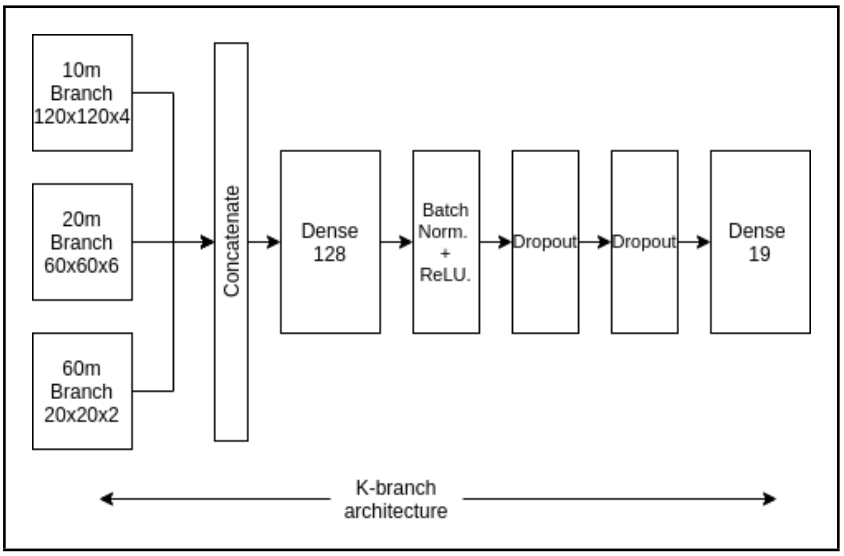


Figure 4.1: K-branch architecture schema diagram

Description of Branch for 10m resolution bands

The input to this branch is of the shape $120x120x4$. The four 10m bands has been stacked together in order to create the input data of this required volume. The 10m branch contains three convolutional layers, with batch normalization layer, ReLU activation layer and dropout layers in between each pair. After the first two convolutional layers (conv1, conv2) of the 10m branch max pool layer has been added. After the last convolutional layers, the output is flattened and a dense layer has been added. The particulars of the layers in the 10m branch are given in the table 4.2. The schematic block diagram of 10m branch has been provided in figure 4.2.

Description of Branch for 20m resolution bands

This branch is similar to 10m branch in design with three convolutional layers, along with batch normalization layer, ReLU activation layer and dropout layers in between. In the 20m branch there is only one Max pool layer in after the first convolutional layer (conv1). As in the 10m branch, (4.2.1), after the last convolutional layers, the output is flattened and a dense layer has been added. The details of the 20m branch are given in the table 4.3. The schematic block diagram of 20m branch has been provided in figure 4.3. The input to the 20m branch has the shape $60x60x6$. This is created by stacking data from the six 20m bands together.

Layer name	Specifications	Input shape	Output shape
Conv1	uses 2D convolution with 32 filters of size (5x5)	120x120x4	120x120x32
MaxPool1	uses pool size (2x2), strides (2x2)	120x120x32	60x60x32
Conv2	uses 2D convolution with 32 filters of size (5x5)	60x60x32	60x60x32
MaxPool2	uses pool size (2x2), strides (2x2)	60x60x32	30x30x32
Conv3	uses 2D convolution with 64 filters of size (3x3)	30x30x32	30x30x64
Flatten	- -	30x30x64	57600
Dense	number of units = 128	57600	128

Table 4.2: Highlights of layers in 10m branch

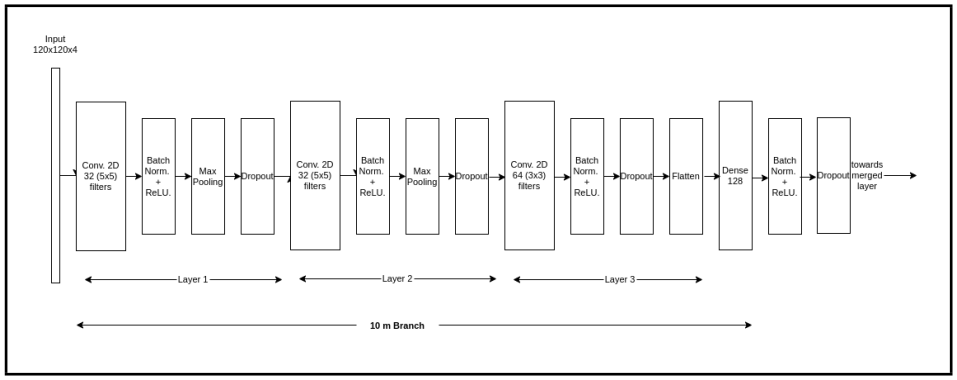


Figure 4.2: 10m branch schema diagram

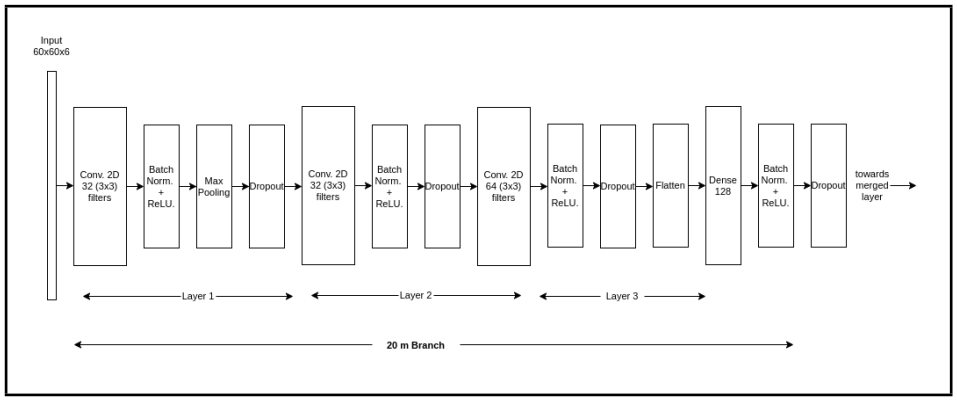


Figure 4.3: 20m branch schema diagram

Layer name	Specifications	Input shape	Output shape
Conv1	uses 2D convolution with 32 filters of size (3x3)	60x60x6	60x60x32
MaxPool1	uses pool size (2x2), strides (2x2)	60x60x32	30x30x32
Conv2	uses 2D convolution with 32 filters of size (3x3)	30x30x32	30x30x32
Conv3	uses 2D convolution with 64 filters of size (3x3)	30x30x32	30x30x64
Flatten	- -	30x30x64	57600
Dense	number of units = 128	57600	128

Table 4.3: Highlights of layers in 20m branch

Description of Branch for 60m resolution bands

Similar to 10m branch (4.2.1) and 20m branch (4.2.1), the 60m branch also has three convolutional layers, with batch normalization layer, ReLU activation layer and dropout layers in between. In this branch the output of the last convolutional layer is flattened and a dense layer has been added as well. The details of the 60m branch are given in the table 4.4. The schematic block diagram of 60m branch has been provided in figure 4.4. The input shape for this branch is $20x20x2$ and has been created by stacking the 60m bands together.

Layer name	Specifications	Input shape	Output shape
Conv1	uses 2D convolution with 32 filters of size (2x2)	20x20x2	20x20x32
Conv2	uses 2D convolution with 32 filters of size (2x2)	20x20x32	20x20x32
Conv3	uses 2D convolution with 32 filters of size (2x2)	20x20x32	20x20x32
Flatten	- -	30x30x64	57600
Dense	number of units = 128	57600	128

Table 4.4: Highlights of layers in 60m branch

Description of the merged architecture

The 10m, 20m and 60m branches are merged together and a classification head was attached to the merged network. Sigmoid activation was applied on the final dense layer to generate the classification result for each class. The schematic block diagram of the merged architecture has been provided in figure 4.1 and details of the layers are given in table 4.5.

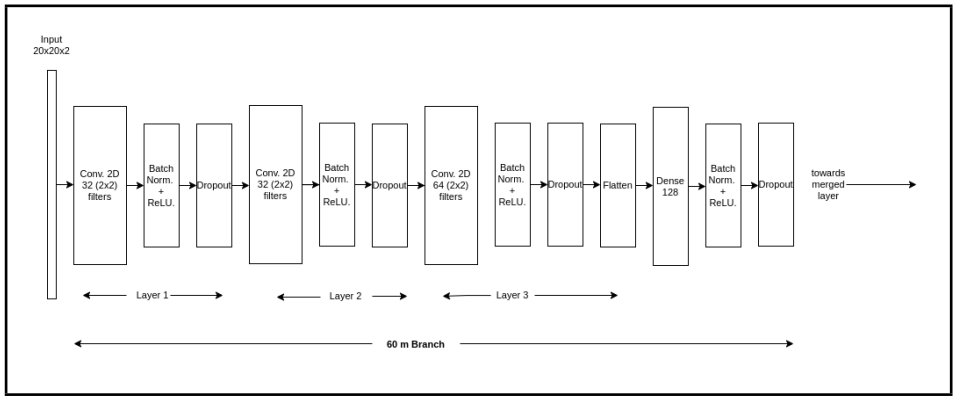


Figure 4.4: 60m branch schema diagram

Layer name	Input shape	Output shape
Concatenate	(128 + 128 + 128) units from three branches	384 units
Dense1	384 units	128 units
Dense2	128 units	19 units

Table 4.5: Highlights of merged K-branch architecture

4.2.2 Architecture for 2D Convolution models with dilation

This section presents the experimental models with 2D convolutions and the modifications added to them. As the second step, experimental models using 2D convolution with dilation are designed based on the 2DConv_base model. A dilation factor=2 has been chosen for this thesis. A higher value has not been chosen keeping in mind the low resolution of the 60m branch images. The dilation=2 has been added to each convolution layer in the all the branches at a time. When dilation is applied to one convolution layer, (for example, the first convolution (conv1) layer of 10m branch), no further changes are introduced anywhere in the model. Thus the remaining layers in the model have exactly the same as the baseline architecture 2DConv_base.

For the sake of readability, the models are broken down and presented branch wise in the tables 4.6, 4.7, 4.8.

4.2.3 Architecture for 3D Convolution model and models with dilated 3D convolution.

Similar to the 4.2.2, this section presents the experimental models with 3D convolutions and the modifications added to them. For these experiments, the 2D convolutions in baseline architecture 2DConv_base. has been first replaced by 3D convolution. As a result, the input dimensions had to be adjusted to the 3 dimensional format. This was done by expanding the dimension of the data by adding an additional axis to the input tensor. This model has been created as a direct 3D counterpart of the baseline architecture 2DConv_base, and has given the alias, 3DConv_base.

Model alias	Branch	Nature of change
10mConv1Dil2	10m branch	Dilation=2 added to the first convolution (conv1) layer of the 10m branch. Remaining network remains unchanged.
10mConv2Dil2	10m branch	Dilation=2 added to the second convolution (conv2) layer of the 10m branch. Remaining network remains unchanged.
10mConv3Dil2	10m branch	Dilation=2 added to the third convolution (conv3) layer of the 10m branch. Remaining network remains unchanged.
10mConvAllDil2	10m branch	Dilation=2 added to conv1, conv2, conv3 layers of 10m branch. Remaining network remains unchanged.

Table 4.6: Models with dilation at 10m branch and 2D convolutions

Model alias	Branch	Nature of change
20mConv2Dil2	20m branch	Dilation=2 added to the second convolution (conv2) layer of 20m branch. Remaining network remains unchanged.
20mConv3Dil2	20m branch	Dilation=2 added to the third convolution (conv3) layer of 20m branch. Remaining network remains unchanged.
20mConvAllDil2	20m branch	Dilation=2 added to conv1, conv2, conv3 layers to 20m branch. Remaining network remains unchanged.

Table 4.7: Models with dilation at 20m branch with 2D convolutions

Model alias	Branch	Nature of change
60mConv2Dil2	60m branch	Dilation=2 added to the second convolution (conv2) layer of 60m branch. Remaining network remains unchanged.
60mConv3Dil2	60m branch	Dilation=2 added to the third convolution (conv3) layer of 60m branch. Remaining network remains unchanged.
60mConvAllDil2	60m branch	Dilation=2 added to the conv2 and conv3 layer of 60m branch. Remaining network remains unchanged.

Table 4.8: Models with dilation at 60m branch with 2D convolutions

This model has been presented in the table 4.9.

Model alias	Nature of changes
3DConv_base	No dilation has been applied. All the 2D convolutions has been changed to 3D convolution across the network. Also, all the max pool layers were updated from 2D to 3D volume. The input volumes had to be updated to 3D volumes as well.

Table 4.9: 3D convolutions base model architecture

Next, multiple variations of this 3D convolution model 3DConv_base are created with dilation=2 applied to each convolution layer in the all the branches at a time. Naturally, when dilation is applied to one convolution layer, (for example, conv1 of 10m branch), the remaining layers remain unchanged and same as 3DConv_base. The list of models are broken down branch-wise and presented in the tables 4.10, 4.11, 4.12.

Model alias	Branch	Nature of changes
3D_10mConv1Dil2	10m branch	3D convolution applied all over the network with dilation=2 added to the first convolution (conv1) layer of 10m branch.
3D_10mConv2Dil2	10m branch	3D convolution applied all over the network with dilation=2 added to the second convolution (conv2) layer of 10m branch.
3D_10mConv3Dil2	10m branch	3D convolution applied all over the network with dilation=2 added to the third convolution (conv3) layer of 10m branch.
3D_10mConvAllDil2	10m branch	3D convolution applied all over the network with dilation=2 added to conv1, conv2, conv3 layers of 10m branch.

Table 4.10: 3D convolution models with dilation=2 at 10m branch

4.3 Experimental Setup

All the models defined in 4.2.2 and 4.2.3 have been used for classification using the datasets described in section 4.1. These experiments are run in the HPC clusters of Technische Universität Berlin, and were run on 1 node with 100GB resident memory per node and 1 Tesla GPU accelerator with nvidia/cuda/10.0 enabled. Software libraries used for the creating the tfrecord datasets, creating the models and running the classification jobs are: Python 3.6, Tensorflow 2.3.0, Keras 2.4.0 and Scikit-learn libraries [78].

Model alias	Branch	Nature of changes
3D_20mConv1Dil2	20m branch	3D convolution applied all over the network with dilation=2 added to the first convolution (conv1) layer of 20m branch.
3D_20mConv2Dil2	20m branch	3D convolution applied all over the network with dilation=2 added to the second convolution (conv2) layer of 20m branch.
3D_20mConv3Dil2	20m branch	3D convolution applied all over the network with dilation=2 added to the third convolution (conv3) layer of 20m branch.
3D_20mConvAllDil2	20m branch	3D convolution applied all over the network with dilation=2 added to conv1, conv2, conv3 layers of 20m branch.

Table 4.11: 3D convolution models with dilation=2 at 20m branch

Model alias	Branch	Nature of changes
3D_60mConv1Dil2	60m branch	3D convolution applied all over the network with dilation=2 added to the first convolution (conv1) layer of 60m branch.
3D_60mConv2Dil2	60m branch	3D convolution applied all over the network with dilation=2 added to the second convolution (conv2) layer of 60m branch.
3D_60mConv3Dil2	60m branch	3D convolution applied all over the network with dilation=2 added to the third convolution (conv3) layer of 60m branch
3D_60mConvAllDil2	60m branch	3D convolution applied all over the network with dilation=2 added to conv1, conv2, conv3 layers of 60m branch

Table 4.12: 3D convolution models with dilation=2 at 60m branch

During the training process, the training parameters had to be carefully adjusted in order to reach optimal classification outcome. The training job as run using learning rates = 0.01, 0.001 and 0.0001 to ascertain the appropriate learning rate. The batch size and number of epoch parameters were also fixed over multiple training attempts. For optimal training duration, the early stop callback has been used. The patience and delta parameters had to be adjusted carefully to ensure the job is stopped only after the training has converged. The parameters used while training the classifier have been presented in the table 4.13

Parameter	Value
Optimizer	Adam optimizer, with learning rate = 0.0001
Loss	Binary Cross Entropy
Number of epochs	100
shuffle buffer size	20000
Early Stopping parameter	monitored parameter='loss', $min\_delta = 1e-4$, patience=5

Table 4.13: Lists of parameters used for classification

For visualizing the results, the best performing models were first shortlisted. This was done by comparing the macro average f1-score of every model. The visualizations were conducted using GPU-enabled Python3 Google compute engine, with 12GB RAM and 107 GB disk. Software libraries used include: python3.6 and Innvestigate [74] with tensorflow 1.15. Innvestigate [74] library provides the implementation of LRP. For visualizing the intermediate activations and results of Grad-CAM, the example codes from the Keras library [79] has been used.

4.4 Evaluation metrics

This section describes the metrics used to evaluate the multi-label classification that forms the first part of the experiments. As discussed in 4.1, BigEarthNet archive comprises of multi-labelled images. Thus, multi-labelled classification metrics has been used in this thesis to evaluate the results.

Evaluation of multi-labelled classification poses some challenges compared to binary or multi-class classification. The classification result for an image is a vector with size n, where n is the number of total labels. In this vector, the predicted value for each individual label can be wrong or right. Thus for any image i, the classification result vector, can be either totally or only partially correct [80].

For this thesis, precision, recall, f1-scores have been chosen as metrics to evaluate the classification result. These measures can be defined as follows:

1. precision: This is the measure of the total predicted correct labels (or true positives (TP)) to the total number of predicted labels (TP + false positives (FP)).This can be represented as:

$$\text{precision} = \frac{True\ Positive}{Total\ Predicted\ labels} \tag{4.1}$$

2. recall: This is the measure of the total predicted correct labels (TP) to the total number of correct labels (TP + false negatives (FN)).This can be represented as:

$$\text{recall} = \frac{True\ Positive}{Actual\ Positive\ labels} \tag{4.2}$$

3. f1-score: This is the weighted average of the precision and recall values [78].

$$\text{f1-score} = 2 * \frac{(precision * recall)}{(precision + recall)} \tag{4.3}$$

However, these metrics can be calculated with respect to either each class label or globally. For the training phase of the models, **micro-precision**, **micro-recall**, and **micro-f1-scores** have been calculated to evaluate the performance per epoch. These metrics are calculated globally without considering class labels. While testing the classification models, however, it is important to evaluate the performance of the models for each class label. Hence, in this phase, micro-precision, micro-recall, and micro-f1-score metrics have been used. To this end, two functions from the scikit-learn library [78] have been used, namely, multi-labelled confusion matrix and classification report.

- The *multi-labelled confusion matrix* treats the multi-labelled problem like multiple binary classification problem, and thus creates one confusion matrix for each class. This provides the true positive (TP), false positive (FP), true negative (TN) and false negative (FN) values for each class label. This has been used to calculate precision, recall and f1-scores per class.

- The *classification report* provides precision, recall and f1-score for each class label, along with micro, macro, weighted and sampled average values for each of the used metrics.

In order to evaluate the classification data, the performance of each class label for each model is considered. As a first step, the classification result for a model is analysed to obtain the highest performing classes for that model. Next, the performances of all the models are compared with respect to the set of these high performing classes. This gives an insight into the correlation between performance of each classes and models. This information helps to narrow down the classes and models of interest and forms the basis for the visualization tasks carried out in this thesis.

5 Results

This chapter presents the results of the classification experiments and the visualizations performed to understand the classification models. This chapter has been arranged into two sections. The first section 5.1, presents a discussion on the classification results. In the second section 5.2, the results of visualization tasks have been presented and analysed.

5.1 Experimental results

This section presents the performance of the classifiers and analyses the results. At the start of this section, a brief discussion on the training result from the 2DConv_base baseline model has been presented. Then, the classification results for the baseline model and some of the top performing models have been presented and compared.

5.1.1 Training result of baseline model

This sub-section presents a brief description of the training performance of the 2DConv_base baseline model. The result obtained from this model has been considered as the benchmark for this thesis. The training parameters were already discussed in the section 4.3. The figure 5.1 shows the trajectory of the training loss curve at each epoch. The training job ran for a total of 63 epochs. The duration was regulated by the early stop callback, which stopped the training process once the variance of loss between epochs slowly reduced to within the acceptable limit. The micro-precision, micro-recall and micro F1-score values at the end of 63 epochs were as follows: (i) micro-precision: 0.1244; (ii) micro-recall: 0.5066 and (iii) micro-F1-score: 0.1922.

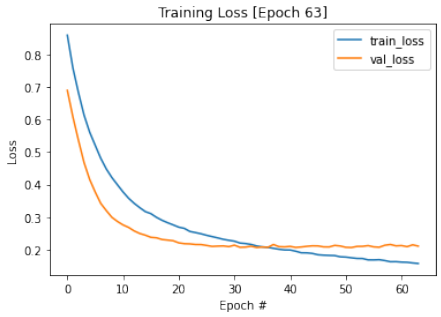


Figure 5.1: Training and validation loss

From the above result, it can be concluded that the model performed reasonably well with the learning rate and the heavily unbalanced data. However, it can be commented that perhaps the

training job should have been stopped a few epochs earlier to avoid any chances of overfitting. Thus a slightly smaller value of patience parameter in the early stop callback would have been a more appropriate choice.

5.1.2 Analysis of classification results

This sub-section presents the classification results from the baseline model as well as several of the top performing models and compares them to gain a more valuable insight into the impact of the model architectures on the class labels. The classification results obtained by evaluating the 2DConv_base baseline model on the test dataset have been presented in table 5.1.

Class	Label	Precision	Recall	F1-score	Support
0	Urban Fabric	0.38135	0.23154	0.28813	17928
1	Industrial or Commercial Units	0.20065	0.4380	0.07191	2808
2	Arable land	0.70547	0.59083	0.64308	47149
3	Permanent crops	0.0000	0.0000	0.0000	6812
4	Pastures	0.34663	0.71059	0.46596	24170
5	Complex cultivation patterns	0.39522	0.31941	0.35329	25638
6	Land principally occupied by agriculture, with significant areas of natural vegetation	0.43500	0.19574	0.26999	32052
7	Agro-forestry areas	0.0000	0.0000	0.0000	7261
8	Broad-leaved forest	0.41918	0.11603	0.18175	34129
9	Coniferous forest	0.60858	0.69993	0.65107	39531
10	Mixed forest	0.60062	0.51046	0.55188	42640
11	Natural grassland and sparsely vegetated areas	0.0000	0.0000	0.0000	2799
12	Moors, heathland and sclerophyllous vegetation	0.08323	0.20083	0.11769	3859
13	Transitional woodland, shrub	0.47902	0.33631	0.39517	36211
14	Beaches, dunes, sands	0.00263	0.04525	0.00497	221
15	Inland wetlands	0.13827	0.29725	0.18875	5349
16	Coastal wetlands	0.0000	0.0000	0.0000	310
17	Inland waters	0.27632	0.00130	0.00258	16177
18	Marine waters	0.68905	0.93209	0.79235	18023

Table 5.1: Classification report of 2DConv_base baseline model

The table 5.1, presents the macro-precision, macro-recall and macro-F1-scores of all the

classes. The support parameter in this table indicates the number of instances of the class that was encountered during the evaluation. From the table 5.1, some observations can be made. They are as follows:

- (i) The classes Permanent crops (class 3), Agro-forestry areas (class 7), Natural grassland and sparsely vegetated areas (class 11) and Coastal wetlands (class 16) have not been detected at all during the classification. The number of samples of these classes is low. However, classes with similar number of samples have been detected by the model. By referring to the confusion matrix for these classes, it was observed that the number of False Positive (FP) instance was equal to the total number of samples for each of these classes. This is the reason why none of the samples belonging to these classes could be correctly identified by the model. This behavior was observed for all other classification models as well.

- (ii) Some of the classes have a higher precision value with a lower recall value. These classes are: Urban Fabric (class 0), Arable land (class 2), Land principally occupied by agriculture, with significant areas of natural vegetation (Class 6), Broad-leaved forest (class 8), Inland waters (class 17). By consulting the confusion matrix for these classes, it has been observed that the False Negative (FN) values for these classes are much higher than the False Positive (FP) predictions. This means that the model can be improved to reduce the number of FN predictions and increase correct identification of these classes.

- (iii) Some of the classes have a lower precision value compared to higher recall value. These classes are: Industrial or Commercial Units (class 1), Pastures (class 4), Moors, heathland and sclerophyllous vegetation (class 12), Beaches, dunes, sands (class 14), Inland waters (class 15) and Marine waters (class 18). Out of these classes, Beaches, dunes, sands had fewer number of samples compared to the others. From the confusion matrix for these individual classes, it can be observed the False Positive (FP) classes are very high compared to the False Negative (FN) predictions. This means that the model can be improved to reduce the number of FP predictions and increase correct identification of these classes.

- (iv) The classes, Complex cultivation patterns (class 5), Coniferous forest (class 9), Mixed forest (class 10), Transitional woodland, shrubs (class 13) have almost equal values for precision and recall. This was supported by the confusion matrix for these classes and this means should be good at identifying these classes.

Based on these above observations, this thesis proposes to compare the class-wise performances of other models and verify if this behavior persists with other models as well or if they are improved by change in architecture. In order to select the overall best performing models, the micro-average F1-score of the baseline model is compared with those of all the experimental models and a set of 11 models were shortlisted. The table 5.2 presents the micro-average F1-scores for the selected models. The complete list of F1-scores of all the experimental models is available at the appendix A.2.

Model alias	Micro-average F1-score
2DConv_base	0.44714
60mConvAllDil2	0.47254
3DConv_base	0.49411
3D_10mConv2Dil2	0.49087
3D_10mConv3Dil2	0.48516
3D_10mConvAllDil2	0.4779
3D_20mConv1Dil2	0.48168
3D_20mConv2Dil2	0.48342
3D_20mConv3Dil2	0.4749
3D_20mConvAllDil2	0.4743
3D_60mConv2Dil2	0.48538
3D_60mConv3Dil2	0.47972

Table 5.2: Micro average F1-score for top performing models

Now, by cross-referencing above class-wise insight from the baseline model and the performances of the shortlisted models given in the table 5.2, the class labels can be grouped into three categories, (i) classes whose F1-score improves compared to baseline 2DConv_base with use of 3D convolution, (ii) classes whose F1-score remained almost unchanged with respect to baseline 2DConv_base with use of 3D convolution, (iii) classes that performed better with 2D convolution.

This section presents the results of the classes whose F1-scores were improved by the models using 3D convolution. These F1-scores of all the classes were compared and the classes that showed higher F1-scores with 3D convolution models were grouped together. The confusion matrices of these models were also compared and it shows definite improvements in True Positive (TP) predictions with reduction in FP and FN vales. The results from the models showing most improvements have been presented in the table 5.3. To adjust with the paucity of space, only the class numbers have been mentioned in this table. The class names can be cross-checked from the table 5.1.

This section presents the second category of classes that remained almost unaffected by use of 3D convolution. For these classes, the F1-score of the models using 2D convolution were similar to those of models using 3D convolution. On comparing the confusion matrices, it was observed that for all of these classes the False Positive (FP) values increased. For class Marine water (class 18), the False Negative (FN) value increased as well, while for classes Coniferous Forest (class 9) and Mixed Forest (class 10) the FN values decreased. These results (with only class numbers) have been presented in the table 5.4. The class names can be cross-checked from the table 5.1.

There was one class which distinctly performed better with the experimental models using 2D convolution. Only the Moors, heathland and sclerophyllous vegetation (class 12) showed this behaviour. On comparing the confusion matrices for this class, the 2D convolution models were found to have significantly better TP value with reduction in FP values as well. The F1-scores for some of the 2D convolutional models have been presented in the table 5.5.

Class	2DConv_-base	3DConv_-base	3D_-10mConvA-llDil2	3D_-20mConv2Dil2	3D_-60mConv2Dil2
Class 0	0.28813	0.36214	0.36887	0.33636	0.29323
Class 1	0.07191	0.1057	0.16811	0.07098	0.08875
Class 5	0.35329	0.43219	0.37453	0.38802	0.41857
Class 6	0.26999	0.35286	0.37187	0.39281	0.36471
Class 8	0.18175	0.36593	0.22231	0.36948	0.36611
Class 13	0.39517	0.37831	0.47786	0.4037	0.37619
Class 14	0.00497	0.01393	0.0295	0.00977	0.01175
Class 15	0.18875	0.18764	0.21411	0.2254	0.18358
Class 17	0.00258	0.03088	0.01797	0.04093	0.0381

Table 5.3: Classes improved by 3D convolution

2DConv_base	20mConv3Dil2	60mConv2Dil2	60mConv3Dil2
0.08323	0.08798	0.09649	0.08686

Table 5.5: Class performing better with 2D convolution

Based on the above discussed observations, a few hypothesis can be formed. The number of instances of each class plays a definite role in the classification result. Few classes showed improvement with use of 3D convolution architecture (i.e. models with as well as without dilation). This could be possible because 3D convolution is good for volumetric analysis and for extraction of low level features. Due to its 3D kernel, 3D convolution is able to extract shapes better. The image patches from the BigEarthNet [1] contains multiple topographic and land-cover signatures. It is possible that the extra dimension of data makes the 3D kernel more capable of identifying land-cover features compared to the 2D convolution. But at the same time, for the classes that did not perform better with 3D convolution, it can be argued that due to the nature of the land-form, 3D convolution cannot detect any additional discernible features.

Regarding the use of dilation, an observation can be made. Dilation increases the size of the receptive field while keeping the output volume same. It helps in discerning features from a crowded scene and for understanding global features. However, with the reduction of spatial resolution, the features in an image patch become coarser. Coarse features would make class-

Class	2DConv_-base	3DConv_-base	3D_-10mConvA-llDil2	3D_-20mConv2Dil2	3D_-60mConv2Dil2
Class 2	0.64308	0.67531	0.68031	0.63046	0.66412
Class 4	0.46596	0.45918	0.4631	0.45604	0.44767
Class 9	0.65107	0.65718	0.66502	0.65205	0.64292
Class 10	0.55188	0.5891	0.53377	0.58857	0.59773
Class 18	0.79235	0.86753	0.87007	0.88301	0.87798

Table 5.4: Classes unchanged by 3D convolution

wise local features highly imperceptible. Thus, it is possible that application of dilation to the lower-resolution bands has significantly reduced the influences of these bands towards the overall classification score. In the subsequent sections, different visualization techniques has been used to further understand these observations.

5.2 Analysing classification models through visualization

In this section different visualization techniques have been used to explain the observations made so far. In the first sub-section 5.2.1, the intermediate activations or feature maps of convolution layers have been visualized to understand the interpretation of features at each layer by the network. Next, in sub-section 5.2.2, LRP method has been used to generate heatmaps that localize regions in an image that correspond to each class. Finally, in sub-section 5.2.3, Grad-CAM has been used to generate heatmaps and obtain the areas of maximum influence for the top predicted class.

5.2.1 Visualizing intermediate activations

In this section, intermediate activations (feature maps) have been generated for the convolution layers, in order to analyse the features observed by the model at each level. The feature maps of the same layer from different architectural models have been compared side-by-side to gain insight into the effects of convolution and dilation on the decision making process of the network. For this task, a sample image has been selected along with four models, which have been used to perform a layer-wise and band-wise analysis. For the first example, the first convolution layers from the 10m bands have been selected. The sample image has been presented in figure 5.2. The classes present in this figure are Arable land (class 2), Complex cultivation patterns (class 5), Land principally occupied by agriculture, with significant areas of natural vegetation (class 6), Transitional woodland, shrub (class 13), Inland waters (class 17). The motivation

behind selection of these models is to show results from 2D convolution, 2D convolution with dilation, 3D convolution and 3D convolution with dilation together to make the comparison more intuitive.

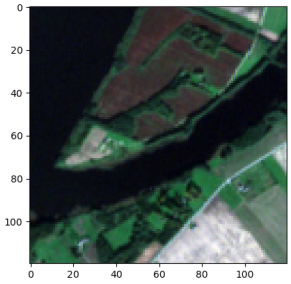
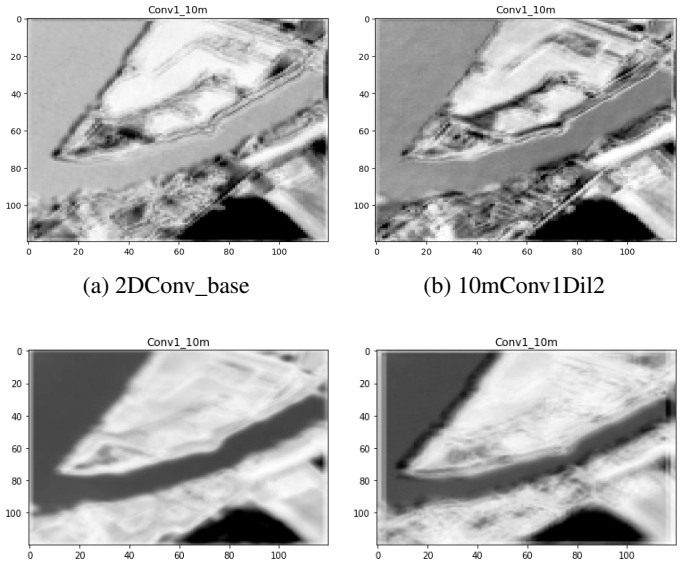


Figure 5.2: Sample image for visualizing intermediate activation

The figure 5.3 presents a side-by-side comparison of the highest activated feature map from **first convolutional layer (conv1) from the 10m band** of the models: 2DConv_base, 10mConv1Dil2, 3DConv_base, 3D_10mConv1Dil2 respectively. This analysis has not been performed class-wise as it is difficult to visually estimate the correct classes, especially when the classes are quite similar.



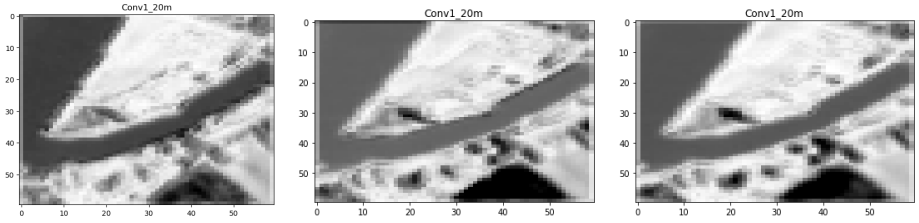
(a) 2DConv_base (b) 10mConv1Dil2

(c) 3DConv_base (d) 3D_10mConv1Dil2

Figure 5.3: Visualization of Convolution 1 from 10m bands of models 2DConv_base, 10mConv1Dil2, 3DConv_base, 3D_10mConv1Dil2 respectively

It can be observed from the figure 5.3, that the feature maps from 3DConv_base, 3D_-10mConv1Dil2 models are indistinct compared to the 2DConv_base, 10mConv1Dil2 models. The 2DConv_base, 10mConv1Dil2 models have been successful in discerning the difference between the types of agricultural landscapes. On comparison with the original image 5.2, it can be observed that these two models were better at identifying the class boundaries. This can be observed by the fine difference in gradient (i.e. change from light to dark colour) noticed in the image along the boundaries. It is also observed that the feature map produced by 10mConv1Dil2 model captures the details better and highlights the class boundaries better than the 2DConv_-base. The 10mConv1Dil2 model is thus the best performing one amongst the chosen models using the first convolution from 10m bands. This is an interesting observation as this suggests that the 2D convolution with dilation=2 was more suited to pick up the details of the features captured in this image.

Next, for the same sample image, the feature maps from the **first convolution layer (conv1) from 20m bands** have been visualized. For this visualization, the chosen models are: 2DConv_-base, 3DConv_base, 3D_20mConv1Dil2.

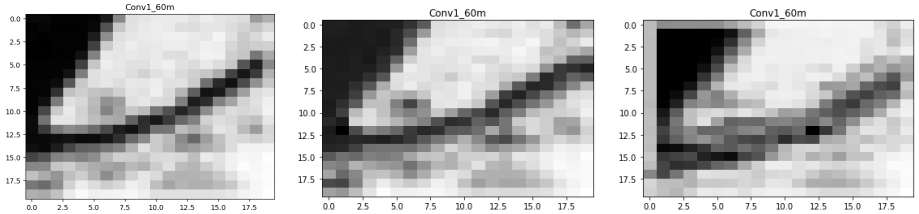


(a) 2DConv_base (b) 3DConv_base (c) 3D_20mConv1Dil2

Figure 5.4: Visualization of Convolution 1 from 20m bands of models 2DConv_base, 3DConv_base, 3D_20mConv1Dil2 respectively

The figure 5.4 gives a mixed result. The feature map from the 2DConv_base model picks up the features in the middle of the image significantly better than the other models. However, 3DConv_base and 3D_20mConv1Dil2 models in turn are able to discern some of the features in the bottom part of the image better than the baseline model. This is an interesting observation and from this observation it can be assumed that the use of 3D convolution has helped in improvement at discerning the structural information of these features. There is no significant difference observed in the feature maps between 3DConv_base and 3D_20mConv1Dil2 models. This is significant because dilation had been applied to the 3D_20mConv1Dil2 model at the first convolution layer in the 20m branch. It could be concluded that even with a dilated kernel, the model has been able to pick up the global features and construct the shape information properly.

The final comparison for first layer has been carried out between the feature maps from the **first convolution layer (conv1) from 60m bands**.For this visualization, the chosen models are: 2DConv_base, 3DConv_base, 3D_60mConv1Dil2.

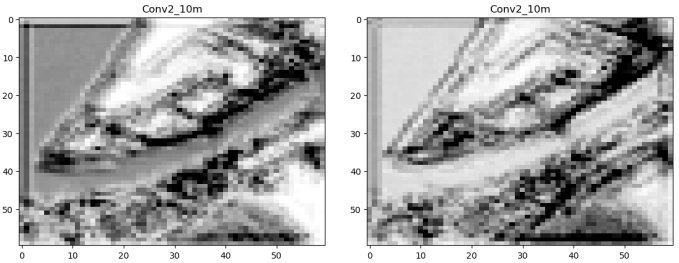


(a) 2DConv_base (b) 3DConv_base (c) 3D_60mConv1Dil2

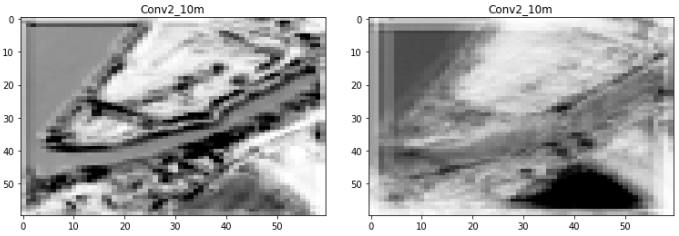
Figure 5.5: Visualization of Convolution 1 from 20m bands of models 2DConv_base, 3DConv_base, 3D_60mConv1Dil2 respectively

As it can be observed from the figures in 5.5, that the 60m samples are very pixelated and course. This does not enable identification of fine-grained features. Only the region representing inland water can be identified due to its distinct nature. Although, these images are not very informative, the change in gradients indicate the boundary of the inland water region with those belonging to other classes. However, no concrete conclusion could be reach based on these feature maps.

The visualization of the **second convolution layer (conv2) from 10m bands** convolutions have been carried out using the the models 2DConv_base, 10mConv2Dil2, 3DConv_base, 3D_-10mConv2Dil2. As with the previous examples for fist convolution layer, the motivation here is to compare side-by-side the effects of 2D convolution, 2D convolution with dilation, 3D convolution and 3D convolution with dilation on the sample image 5.2.



(a) 2DConv_base (b) 10mConv2Dil2



(c) 3DConv_base (d) 3D_10mConv2Dil2

Figure 5.6: Visualization of Convolution 2 from 10m bands of models 2DConv_base, 10mConv2Dil2, 3DConv_base, 3D_-10mConv2Dil2 respectively

In the figure 5.6 it can be observed that the feature map from the model 3D_10mConv2Dil2, which uses 3D convolution and dilation, to be the most indistinct. Out of the remaining feature maps, it can be observed that the boundaries between areas of contrasting activations have been produced very distinctly in 3DConv_base. Since, this is the second layer convolution, the features are more global in nature compared to those of the first layer. This can be observed in the feature map for 2DConv_base and 10mConv2Dil2. Out of the four feature maps, it can be concluded that the 3D convolution model captured the most information and produced the most distinct features.

The feature maps for second convolutions layers in 20m and 60m branches as well as the those from the layer 3 convolutions from 10m,20m and 60m branches, were too pixelated and indistinct to be of any use towards meaningful interpretation. Thus, those layers had to be excluded from this study.

Overall, visualization of feature maps or internal activations of a convolution layer does provide essential visual clues towards the internal representation of the images and can help in

identifying shapes identified from one layer to another in a gradual hierarchy. It has been observed that the 3D convolution was able to capture the structural information of smaller sections of the image better than the 2D convolution model. This inference holds for both 10m and 20m bands. This observation supports patterns noticed in some of the classes where the results improved with application of 3D convolution.

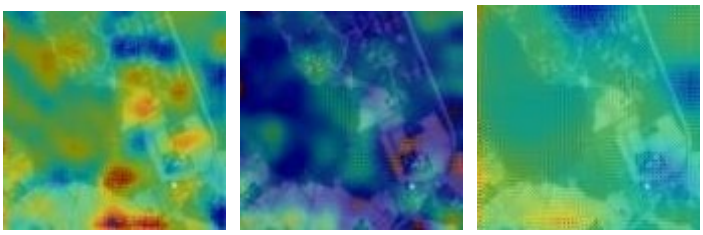
5.2.2 Visualization using LRP

In this sub-section, LRP visualization has been applied to the different network architectures in order to visualize regions belonging to predicted classes. LRP [13] was discussed in the section 2.3.3. In this section, the Innvestigate library [74] has been used to apply LRP to the top performing models. For the purpose of visualization , the Sequential_preset_a_flat LRP has been used with parameters epsilon=1, and neuron_selection_mode="index". This visualization is applied only on some of the top performing models and compared based on the heatmap generated for the predicted class labels. In many cases, even when the classifier predicted the class labels correctly, LRP was unable to calculate any heatmap output for one or more of the correct class labels. A significant benefit provided by LRP class is that it enables the visualization of the exact area or feature that caused the classifier to wrongly classify one class as another. Thus, using LRP it is possible to understand the cause behind misclassification of a class, which can facilitate the improvement of model performance.

The visualizations are carried out using only the models 2DConv_base (baseline) model, 3DConv_base and 3D_10mConvAllDil2 models. 3DConv_base and 3D_10mConvAllDil2 models were some of the highest performing models and had showed good F1-scores for multiple classes. The motivation behind this selection was to compare the class-wise localization capability for models using 2D convolution, 3D convolution and 3D convolution with dilation.

Figure 5.7: Sample image for visualizing LRP heatmap

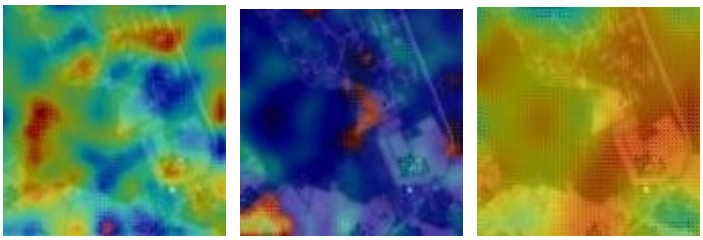
The sample image was annotated with six class labels which include: Urban fabric (class 0), Arable land (class 2), Land principally occupied by agriculture, with significant areas of natural vegetation (class 6), Coniferous forest (class 9), Mixed forest (class 10), and Inland water (class 17). For this thesis, the heatmaps for all these classes has been visualized for the three chosen models.



(a) 2DConv_base (b) 3DConv_base (c) 3D_10mConvAllDil2

Figure 5.8: Visualization of heatmaps for class urban fabric for models 2DConv_base, 3DConv_base, 3D_10mConvAllDil2 respectively

For the class: Urban fabric (class 0), the heatmaps are depicted in 5.8a. It is clearly visible, that the 2DConv_base model produced a better response for this class. In the heatmap 5.8a, images belonging to urban fabric class has been annotated, although some additional areas have been highlighted as well. The 3DConv_base model generates a very dim response towards the location of the class as seen in the heatmap 5.8b. 3D_10mConvAllDil2 does not produce any meaningful localization for the class at all. This is an interesting observation, because based on the classification results, the performance of class urban fabric had been observed to have improved with use of 3D convolution. It is possible that the area occupied by the class is small and the use of 3D convolution hampers the model from learning the shape of this area, compared to 2D convolution. However, it can also be observed that this result could be a solitary instance and not representative of the LRP performance for class urban fabric across all models.

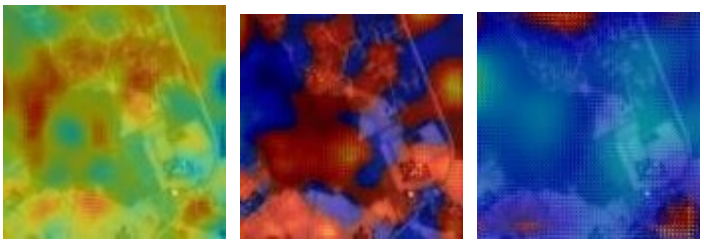


(a) 2DConv_base (b) 3DConv_base (c) 3D_10mConvAllDil2

Figure 5.9: Visualization of heatmaps for class arable land for models 2DConv_base, 3DConv_base, 3D_10mConvAllDil2 respectively

For the **class Arable Land (class 2)**, the heatmaps have been displayed in the figure 5.9. From these figures it can be observed that the 2DConv_base model predicted the class correctly, but while localizing the areas, many regions that do not belong to this class have been annotated as well. The heatmap generated by 3DConv_basemodel, is very pronounced but annotates only

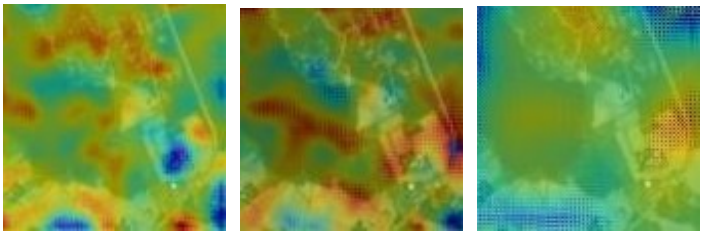
a small area belonging to this class and misses the rest. It however, does not annotate any unnecessary regions. This could possibly indicate that the 3D convolution learnt the shapes associated with the class properly and could successfully identify it in this sample image. In case of 3D_10mConvAllDil2, the heatmap generated covers most of the area belonging to class arable land, however the heatmap is not distinct and appears scattered. This behavior could possibly be attributed to the dilation of the convolution layers in this model. Although the identification of the related area is correct, it is possible the dissipated heatmap represents the influence of dilation.



(a) 2DConv_base (b) 3DConv_base (c) 3D_10mConvAllDil2

Figure 5.10: Visualization of heatmaps for class Land principally occupied by agriculture, with significant areas of natural vegetation (class 6) for models 2DConv_base, 3DConv_base, 3D_10mConvAllDil2 respectively

The class **Land principally occupied by agriculture, with significant areas of natural vegetation (class 6)** was successfully identified by all the classifiers. However, the localization results were less than satisfactory. Based on the figure 5.10a, it can be observed that the heatmap generated by 2DConv_base has an overlap with the heatmap for class arable land (5.9a). This could be explained by the fact that both these classes represent similar features related to vegetation and thus the regions could have an overlap. In case of the 3DConv_base model, distinct regions were identified and localized correctly. The 3D_10mConvAllDil2 model however, did not produce any meaningful heatmap for this class at all.



(a) 2DConv_base (b) 3DConv_base (c) 3D_10mConvAllDil2

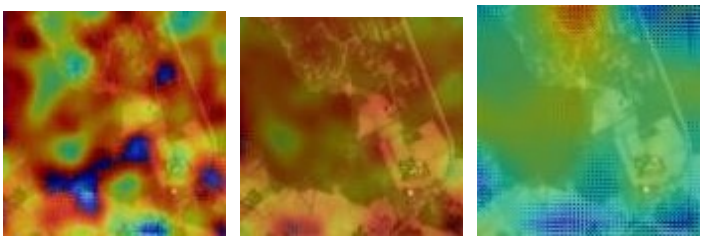
Figure 5.11: Visualization of heatmaps for class Coniferous forest (class 9) for models 2DConv_base, 3DConv_base, 3D_10mConvAllDil2 respectively

From the heatmaps for **Coniferous forest (class 9)** in figure 5.11, a definite improvement from the 2D convolution to 3D convolution can be observed. Although the models 2DConv_base and 3DConv_base localized the same areas for this class, the areas localized in figure 5.11b are much more pronounced compared to the figure 5.11a. The 3D_10mConvAllDil2 model identifies similar areas as well from the sample image, however, they appear less distinct and dissipated. As already stated, this could be the effect of dilation. Thus, although it identifies the correct areas, it produces a dilated heatmap of the area of influence.

The heatmaps for classes Mixed forest (class 10) and Inland water (class 17) have been displayed in the figures 5.12 and 5.13 respectively. It can be seen from these two set of figures that all the models failed in generating meaningful heatmaps for these two classes. This can be supported by the fact that only the model 3DConv_base was able to predict these two classes. However, the lack of distinct heatmap for the 3DConv_base for these two classes could not be explained. As a final observation for visualization using LRP, it can be said that LRP is effective in class-wise localization of distinct areas of the image. However, there are instances where it fails to localize classes that has been detected by the classifier. The biggest advantage of LRP is the fact that it is model agnostic and can be implemented on any architecture.

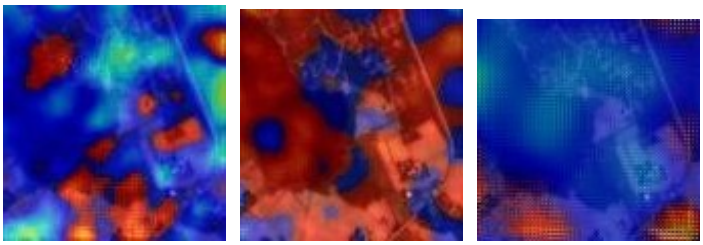
5.2.3 Visualization using Grad-CAM

In this sub-section, Grad-CAM technique has been applied on some of the architectural models to generate a heatmap for the areas of maximum influence on the prediction result. The concept behind Grad-CAM [14] has been discussed in the section 2.3.3. For visualizing classifier performance using Grad-CAM, Keras [79] library has been used. Grad-CAM calculates the significance of the highest activated neuron with respect to the feature map of the final convolution layer. However, this places a particular constraint on the model architecture. Grad-CAM expects that the final convolution layer in the model will be followed immediately by the classification layer. This condition however, could not be satisfied by the K-Branch architecture. Thus, Grad-CAM has been applied for visualizations of the K-branch architecture, but with some limitations. These limitations are that Grad-CAM could not be used to generate heatmaps for 3D convolu-



(a) 2DConv_base (b) 3DConv_base (c) 3D_10mConvAllDil2

Figure 5.12: Visualization of heatmaps for class Mixed forest (class 10) for models 2DConv_base, 3DConv_base, 3D_10mConvAllDil2 respectively



(a) 2DConv_base (b) 3DConv_base (c) 3D_10mConvAllDil2

Figure 5.13: Visualization of heatmaps for class Inland water (class 17) for models 2DConv_base, 3DConv_base, 3D_10mConvAllDil2 respectively

tions due to mismatch in expected gradient shape, and it could be used to generate heatmap only for the top predicted class. Thus, visualizations of only 2D convolution models have been performed with Grad-CAM.

With these constraints, the Grad-CAM visualization technique was applied on the 2DConv_base and 60mConvAllDil2 models. The 60mConvAllDil2 model was chosen as this had shown improved performance compared to the 2DConv_base. It had micro F1-score 0.47254 compared to 0.44714 of 2DConv_base model. The motivation has been to compare performance of 2D convolution with dilated 2D convolution.

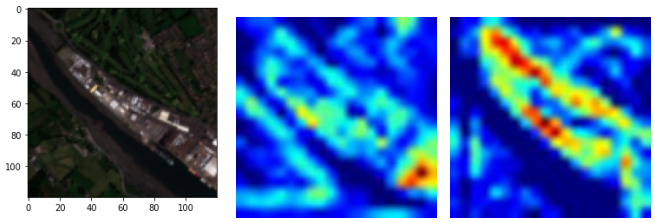
Figure 5.14 presents the comparative heatmap between the two selected models for the **class urban fabric (class 0)**. Although the both the models were able to identify and localize the distinct area for urban fabric in the image, the heatmap generated by the 60mConvAllDil2 model was more localized and intense. It also follows the shape of the exact structure that was highlighted.

Figure 5.15 depicts the original image and heatmaps generated for the **class Coniferous forest (class 9)**. It has been observed that the heatmap for 60mConvAllDil2 is more concise and also

shows higher intensity. It localizes the exact locations that belong to this class and does not highlight any additional regions. The 2DConv_base model however, highlights some areas that do not belong to the class coniferous forest.

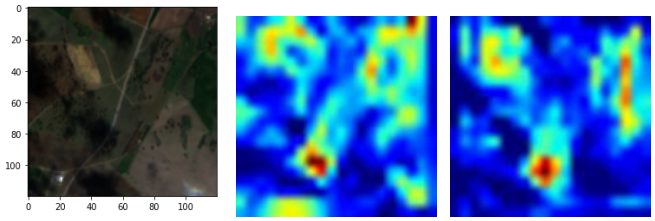
In the figure, 5.16, heatmaps for the **class Arable land (class 2)** has been shown. It can be observed, that the heatmap for 2DConv_base performs better at localizing the area of interest. In this case, the heatmap for 60mConvAllDil2, identifies some areas in the center of the image that are not an exact match with the class. This also supports the earlier observation about Arable land.

Base on these heatmaps, it was observed that Model 60mConvAllDil2 generally performed better in localizing the classes in an image than 2DConv_base model. An explanation for this behaviour could be that the dilation applied to the 60m branches helped in identifying global features that contributed to the better localization of the class. In this case, visualization of the feature maps of the 60m bands, would have proved more insightful into this behaviour. As a final observation it can be concluded that Grad-CAM is very effective in localizing the exact areas of the image that belongs to a specific class. However, the architectural constraint makes this technique difficult to use with models like K-branch.



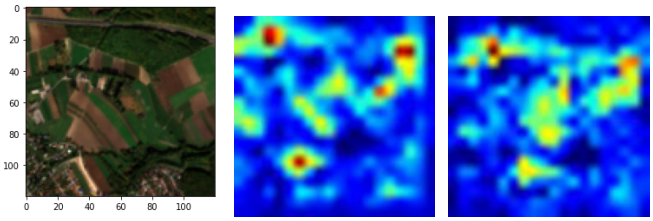
(a) Sample image (b) 2DConv_base (c) 60mConvAllDil2

Figure 5.14: Original sample and heatmap Visualization of class Urban fabric using models 2DConv_base, 60mConvAllDil2 respectively



(a) Sample image 2 (b) 2DConv_base (c) 60mConvAllDil2

Figure 5.15: Original sample and heatmap Visualization of class Coniferous forest using models 2DConv_base, 60mConvAllDil2 respectively



(a) Sample image 4 (b) 2DConv_base (c) 60mConvAllDil2

Figure 5.16: Original sample and heatmap Visualization of class Arable land using models 2DConv_base, 60mConvAllDil2 respectively

Based on all the classification results and the visualizations performed, it can be concluded that understanding the internal representation of a class label in a convolution layer is a complex task. Using techniques like LRP, Grad-CAM and feature map visualization, some insight can be obtained into the decision-making process. In some cases these insights are very clear and intuitive, but for some sample images or classes, the performance can be questionable.

6 Conclusion and Discussion

The goal of this master thesis was to understand the impact of different convolutional parameters. To this end, 2D convolution, 3D convolution and dilation were chosen as the characteristics to be analysed in this study. This thesis was an exploratory study towards understanding and explaining the influences of these convolutional characteristics in multilabel classification of remote sensing images. In order to explain the decision making process of CNNs, different visualization techniques were used. This chapter presents the observation, insights and concluding remarks for this study.

6.1 Observations

In order to conduct this comparative study a baseline architecture was necessary. In this thesis, the K-branch architecture, introduced in [2] and described in [1], was chosen as a guideline for the baseline architecture. The baseline architecture (2DConv_base) was designed using 2D convolution and several adaptations of this model were designed using dilated 2D convolution, 3D convolution and dilated 3D convolution, as already discussed in 4.2. The K-Branch architecture comprises of three input branches accepting inputs of three different dimensions. These models were used to classify the multilabel dataset from BigEarthNet [1] archive.

Based on the classification results of the experimental models it was observed that the network models using 3D convolution architecture (with or without dilation) in general, had a better performance than the 2D convolution models. In order to compare the overall performance of the models, the micro-average F1-score values were used. Based on these F1-scores, a set of high performing network models could be chosen.

On performing a class-wise performance comparison of the models, it was observed that it could be divided into three distinct groups. A group of classes showed definite improvement in performance when models with 3D convolution were used for classification. Another group showed no change in performance from 2D convolution to 3D convolution architecture. These class groups are provided below:

- (i) Classes with improved performance using 3D convolution: Urban fabric (class 0), Industrial or Commercial Units (class 1), Complex cultivation patterns (class 5), Land principally occupied by agriculture, with significant areas of natural vegetation (class 6), Broad-leaved forest (class 8), Transitional woodlands, shrubs (class 13), Beaches, dunes and sands (class 14), Inland water (class 17).

- (ii) Classes with no significant change in performance from 2D to 3D convolution: Arable lands (class 2), Pastures (class 4), Coniferous forest (class 9), Mixed forest (class 10), Marine water (class 18).

- Only Moors, heathlands and sclerophyllous vegetation (class 12) showed improved performance with dilated 2D convolutions.

In order to explain these observations, the feature maps of convolutional layers were visualized. The visualizations were performed layer-wise for each band for a given sample image. This visualization technique was very insightful and it could be observed how the features were observed by the CNN at each layer. A side-by-side comparison between the insight of a 2D convoluted layer, dilated 2D convoluted layer, 3D convoluted layer and dilated 3D convoluted layers were presented. The observations can be listed as follows:

- At first convolution layer of 10m branch, 2D convolution models produced more distinct and distinguishable feature maps compared to 3D convolution. They were able to capture the class changes and smaller features as well.

- It was also observed that feature map from 10mConv1Dil2 model showed improved distinction of class boundaries compared to 2DConv_base baseline model.

- For first convolution layer in 20m branch similar observations were made. It was additionally observed that the 3D convolution models identified some features that the baseline model was unable to identify.

- No particular differences was observed between insights gathered by feature maps from 3DConv_base and 3D_20mConv1Dil2 models. The explanation for this behavior could be that despite a dilated kernel, the model was able to discern global features and construct the essential class boundaries.

- From the feature maps of second convolution from 10m branch, it was observed that, 3D convolutions produced more distinguishable feature maps. In these feature maps the class boundaries were clearer, difference between smaller objects were also visually perceptible. From the observations, it could be concluded that the 3D convolution performed superior to 2D convolution in the second layer of the network.

Overall, it could be concluded that visualizing feature maps is an effective strategy to get valuable insights and helps understand how the features evolve with deeper layers in a CNN. It can also help in understanding which classes are more distinctly captured by specific convolution types.

Next visualization technique used was LRP that helped localize the classes in a multilabel image. For this thesis, all the classes in a sample image were visualized using LRP. Performance of LRP varied on the model used and the class being visualized. The observations are as follows:

- It was observed using LRP visualization that the baseline model architecture was better at identifying areas belonging to Urban fabric class.

- For class arable land, the baseline model performed well, however it highlighted additional regions.

- 3D convolution models were successful in identifying areas belonging to class Arable land. However, 3DCond_base has the best performance as it localized very distinct regions and did not highlight any additional regions in its heatmap.
- For Land principally occupied by agriculture, with significant areas of natural vegetation (class 6), it was observed that although all the models identified some of the relevant areas, there was a lot of overlap between areas belonging to this and Arable Land class. This indicates either better class annotation strategy could be useful or a more in-depth feature extractor could be used for such cases.
- In some cases it was observed that LRP could not generate distinct heatmaps for a class that the classifier predicted.

Overall, it can be concluded that LRP is a very effective strategy for visualizing the exact area that influenced a classifier's decision. It thus helps in explaining the prediction of a CNN. LRP is also model agnostic and can be thus applied to any kind of model architecture. The final visualization technique used was Grad-CAM. This could only be applied for 2D convolution network models. Grad-CAM was also very effective in visualising the regions that influenced the prediction of a class. However, in his thesis it could only be used to visualise the top predicted class. Using sample images, Grad-CAM could successfully localize the areas responsible for prediction of Arable class, Urban fabric and Coniferous forests.

It can be concluded that all these strategies were effective in giving an insight into the behavior of a CNN. A more in-depth study analysing all the class for all the top network models could provide more insight into the internal decision making process of a CNN.

6.2 Challenges faced

The study involved handling large volume of data for the dataset preparations. This made the use of GPU enabled processors necessary. For this thesis there was along training process. The training process had to be calibrated based on multiple parameters, like batch size, learning rate and early stop callback parameters. This process was both time intensive and needed hours of computation. The data output from the different models had to be analysed systematically and meaningfully to select the top performing classes as well as in obtaining the list of models that performed well for each architecture. Apart from these, different libraries for Grad-CAM were attempted to be used for this project. However, most of these were not compatible with the model architecture used in this thesis. As a result, Grad-CAM technique could only be used to visualize the 2D Convolution models and not for the 3D convolution models in this thesis.

6.3 Future work

Multiple ideas for future work can be suggested. It would be very interesting to view more visualizations of classes and models to gather clues for improved architectural decisions. It would also be interesting to see the impact of a dilation value (i.e. other than 2) on these classes.

Apart from dilation, different values of strides can also be used to evaluate the performance of a CNN and gain valuable insights.

Bibliography

[1] G. Sumbul, J. Kang, T. Kreuziger, F. Marcelino, H. Costa, P. Benevides, M. Caetano, and B. Demir, "Bigearthnet dataset with a new class-nomenclature for remote sensing image understanding," *arXiv: Computer Vision and Pattern Recognition*, 2020.

[2] G. Sumbul and B. Demir, "A novel multi-attention driven system for multi-label remote sensing image classification," in *IGARSS 2019 - 2019 IEEE International Geoscience and Remote Sensing Symposium*, 2019, pp. 5726–5729. DOI: 10.1109/IGARSS.2019.8898188.

[3] Z. Zhao, J. Li, Z. Luo, J. Li, and C. Chen, "Remote sensing image scene classification based on an enhanced attention module," *IEEE Geoscience and Remote Sensing Letters*, pp. 1–5, 2020. DOI: 10.1109/LGRS.2020.3011405.

[4] V. H. Phung, E. J. Rhee, *et al.*, "A high-accuracy model average ensemble of convolutional neural networks for classification of cloud image patches on small datasets," *Applied Sciences*, vol. 9, no. 21, p. 4500, 2019.

[5] R. Geirhos, P. Rubisch, C. Michaelis, M. Bethge, F. A. Wichmann, and W. Brendel, "Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness," *CoRR*, vol. abs/1811.12231, 2018. arXiv: 1811.12231. [Online]. Available: http://arxiv.org/abs/1811.12231.

[6] Y. Wang and Y. Wu, "Scene classification with deep convolutional neural networks," 2014.

[7] A. Narayanan, I. Dwivedi, and B. Dariush, "Dynamic traffic scene classification with space-time coherence," *CoRR*, vol. abs/1905.12708, 2019. arXiv: 1905.12708. [Online]. Available: http://arxiv.org/abs/1905.12708.

[8] G. A. Fricker, J. D. Ventura, J. A. Wolf, M. P. North, F. W. Davis, and J. Franklin, "A convolutional neural network classifier identifies tree species in mixed-conifer forest from hyperspectral imagery.," *Remote Sensing*, vol. 11, no. 19, p. 2326, 2019. [Online]. Available: http://dblp.uni-trier.de/db/journals/remotesensing/remotesensing11.html#FrickerVWNDF19.

[9] Y. Li, X. Zhang, and D. Chen, "Csrnet: Dilated convolutional neural networks for understanding the highly congested scenes," *CoRR*, vol. abs/1802.10062, 2018. arXiv: 1802.10062. [Online]. Available: http://arxiv.org/abs/1802.10062.

[10] X. Cui, K. Zheng, L. Gao, B. Zhang, D. Yang, and J. Ren, "Multiscale spatial-spectral convolutional network with image-based framework for hyperspectral imagery classification," *Remote. Sens.*, vol. 11, p. 2220, 2019.

[11] S. Ji, C. Zhang, A. Xu, Y. Shi, and Y. Duan, "3d convolutional neural networks for crop classification with multi-temporal remote sensing images.," *Remote Sensing*, vol. 10, no. 1, p. 75, 2018. [Online]. Available: http://dblp.uni-trier.de/db/journals/remotesensing/remotesensing10.html#JiZXSD18.

[12] W. Samek and K.-R. Müller, "Towards explainable artificial intelligence," in *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, W. Samek, G. Montavon, A. Vedaldi, L. K. Hansen, and K.-R. Müller, Eds. Cham: Springer International Publishing, 2019, pp. 5–22, ISBN: 978-3-030-28954-6. DOI: 10.1007/978-3-030-28954-6_1. [Online]. Available: https://doi.org/10.1007/978-3-030-28954-6_1.

[13] S. Bach, A. Binder, G. Montavon, F. Klauschen, K. Müller, and W. Samek, "On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation," *PLoS ONE*, vol. 10, 2015.

[14] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," *International Journal of Computer Vision*, vol. 128, no. 2, 336–359, 2019, ISSN: 1573-1405. DOI: 10.1007/s11263-019-01228-7. [Online]. Available: http://dx.doi.org/10.1007/s11263-019-01228-7.

[15] M. D. Zeiler and R. Fergus, *Visualizing and understanding convolutional networks*, 2013. arXiv: 1311.2901 [cs.CV].

[16] N. R. Canada, *Government of canada*, 2015. [Online]. Available: https://www.nrcan.gc.ca/maps-tools-publications/satellite-imagery-air-photos/remote-sensing-tutorials/fundamentals-remote-sensing-introduction/9363.

[17] C. S. Investigations, *Energy: The driver of climate*, 2016. [Online]. Available: http://www.ces.fau.edu/nasa/module-2/radiation-sun.php.

[18] *What is remote sensing?* 2020. [Online]. Available: https://earthdata.nasa.gov/learn/backgrounders/remote-sensing.

[19] *Resolutions*. [Online]. Available: https://sentinel.esa.int/web/sentinel/user-guides/sentinel-2-msi/resolutions.

[20] *Satellite resolution*. [Online]. Available: https://www.usna.edu/Users/oceano/pguth/md_help/html/satb7q9a.htm.

[21] *Msi-instrument*. [Online]. Available: https://sentinel.esa.int/web/sentinel/technical-guides/sentinel-2-msi/msi-instrument.

[22] Sinergise, *Simple rgb composites (sentinel-2)*. [Online]. Available: https://custom-scripts.sentinel-hub.com/custom-scripts/sentinel-2/composites/.

[23] CRISP, *Interpreting optical remote sensing images*, 2001. [Online]. Available: https://crisp.nus.edu.sg/~research/tutorial/opt_int.htm.

[24] Sentinel-Hub, *Sentinel-hub/custom-scripts*. [Online]. Available: https://github.com/sentinel-hub/custom-scripts/tree/master/sentinel-2/bands.

[25] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. The MIT Press, 2016, ISBN: 0262035618.

[26] F. Chollet, *Deep Learning with Python*, 1st. USA: Manning Publications Co., 2017, ISBN: 1617294438.

[27] F.-F. Li, A. Karpathy, and J. Johnson, "Cs231n: Convolutional neural networks for visual recognition 2016," [Online]. Available: http://cs231n.stanford.edu/.

[28] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," in *Proceedings of the IEEE*, 1998, pp. 2278–2324.

[29] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds., Curran Associates, Inc., 2012, pp. 1097–1105. [Online]. Available: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf.

[30] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, cite arxiv:1409.1556. [Online]. Available: http://arxiv.org/abs/1409.1556.

[31] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, *Going deeper with convolutions*, cite arxiv:1409.4842, 2014. [Online]. Available: http://arxiv.org/abs/1409.4842.

[32] K. He, X. Zhang, S. Ren, and J. Sun, *Deep residual learning for image recognition*, cite arxiv:1512.03385 Comment: Tech report, 2015. [Online]. Available: http://arxiv.org/abs/1512.03385.

[33] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1251–1258. [Online]. Available: http://openaccess.thecvf.com/content_cvpr_2017/html/Chollet_Xception_Deep_Learning_CVPR_2017_paper.html.

[34] G. E. Hinton, *Rectified linear units improve restricted boltzmann machines vinod nair*.

[35] A. Deshpande, *A beginner's guide to understanding convolutional neural networks part 2*, 2016. [Online]. Available: https://adeshpande3.github.io/adeshpande3.github.io/A-Beginner's-Guide-To-Understanding-Convolutional-Neural-Networks-Part-2/.

[36] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, and Y. Bengio, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, 1929.

[37] S. Ioffe and C. Szegedy, *Batch normalization: Accelerating deep network training by reducing internal covariate shift*, 2015. arXiv: 1502.03167 [cs.LG].

[38] Vdumoulin, *Vdumoulin/conv$_a$rithmetic*, 2016. [Online]. Available: https://github.com/vdumoulin/conv_arithmetic.

[39] S. Bansal, *3d convolutions : Understanding use case*, 2019. [Online]. Available: https://www.kaggle.com/shivamb/3d-convolutions-understanding-use-case.

[40] M. Lin, Q. Chen, and S. Yan, *Network in network*, 2014. arXiv: 1312.4400 [cs.NE].

[41] K. Bai, *A comprehensive introduction to different types of convolutions in deep learning*, Feb. 2019. [Online]. Available: https://towardsdatascience.com/a-comprehensive-introduction-to-different-types-of-convolutions-in-deep-learning-669281e58215.

[42] F. Yu and V. Koltun, *Multi-scale context aggregation by dilated convolutions*, 2016. arXiv: 1511.07122 [cs.CV].

[43] D. Phiri, M. Simwanda, S. Salekin, V. R. Nyirenda, Y. Murayama, and M. Ranagalage, "Sentinel-2 data for land cover/use mapping: A review," *Remote Sensing*, vol. 12, no. 14, p. 2291, Jul. 2020, ISSN: 2072-4292. DOI: 10.3390/rs12142291. [Online]. Available: http://dx.doi.org/10.3390/rs12142291.

[44] R. Pires de Lima and K. Marfurt, "Convolutional neural network for remote-sensing scene classification: Transfer learning analysis," *Remote Sensing*, vol. 12, no. 1, p. 86, Dec. 2019, ISSN: 2072-4292. DOI: 10.3390/rs12010086. [Online]. Available: http://dx.doi.org/10.3390/rs12010086.

[45] G. A. Fricker, J. D. Ventura, J. A. Wolf, M. P. North, F. W. Davis, and J. Franklin, "A convolutional neural network classifier identifies tree species in mixed-conifer forest from hyperspectral imagery," *Remote Sensing*, vol. 11, no. 19, p. 2326, 2019, ISSN: 2072-4292. DOI: 10.3390/rs11192326. [Online]. Available: http://dx.doi.org/10.3390/rs11192326.

[46] R. Stivaktakis, G. Tsagkatakis, and P. Tsakalides, "Deep learning for multilabel land cover scene categorization using data augmentation," *IEEE Geoscience and Remote Sensing Letters*, vol. 16, no. 7, pp. 1031–1035, 2019. DOI: 10.1109/LGRS.2019.2893306.

[47] S. Ji, C. Zhang, A. Xu, Y. Shi, and Y. Duan, "3d convolutional neural networks for crop classification with multi-temporal remote sensing images," *Remote Sensing*, vol. 10, no. 2, p. 75, 2018, ISSN: 2072-4292. DOI: 10.3390/rs10010075. [Online]. Available: http://dx.doi.org/10.3390/rs10010075.

[48] N. Kussul, M. Lavreniuk, S. Skakun, and A. Shelestov, "Deep learning classification of land cover and crop types using remote sensing data," *IEEE Geoscience and Remote Sensing Letters*, vol. 14, no. 5, pp. 778–782, 2017. DOI: 10.1109/LGRS.2017.2681128.

[49] S. Ji, W. Xu, M. Yang, and K. Yu, "3d convolutional neural networks for human action recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 221–231, 2013. DOI: 10.1109/TPAMI.2012.59.

[50] H. Vu, H. Kim, and J. Lee, "3d convolutional neural network for feature extraction and classification of fmri volumes," in *2018 International Workshop on Pattern Recognition in Neuroimaging (PRNI)*, 2018, pp. 1–4. DOI: 10.1109/PRNI.2018.8423964.

[51] R. Hamaguchi, A. Fujita, K. Nemoto, T. Imaizumi, and S. Hikosaka, *Effective use of dilated convolutions for segmenting small object instances in remote sensing imagery*, 2017. arXiv: 1709.00179 [cs.CV].

[52] Y. Li, X. Zhang, and D. Chen, *Csrnet: Dilated convolutional neural networks for understanding the highly congested scenes*, 2018. arXiv: 1802.10062 [cs.CV].

[53] R. R. Devaram, D. Allegra, G. Gallo, and F. Stanco, "Hyperspectral image classification via convolutional neural network based on dilation layers," in *Image Analysis and Processing – ICIAP 2019*, E. Ricci, S. Rota Bulò, C. Snoek, O. Lanz, S. Messelodi, and N. Sebe, Eds., Cham: Springer International Publishing, 2019, pp. 378–387, ISBN: 978-3-030-30642-7.

[54] D. Ciresan, U. Meier, J. Masci, and J. Schmidhuber, "A committee of neural networks for traffic sign classification," in *IN INTERNATIONAL JOINT CONFERENCE ON NEURAL NETWORKS*.

[55] C. Lu and X. Tang, "Surpassing human-level face verification performance on lfw with gaussianface," Tech. Rep., 2014.

[56] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015, ISSN: 14764687. DOI: 10.1038/nature14236. [Online]. Available: https://doi.org/10.1038/nature14236.

[57] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis, "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, pp. 484–, Jan. 2016. [Online]. Available: http://dx.doi.org/10.1038/nature16961.

[58] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. van den Driessche, T. Graepel, and D. Hassabis, "Mastering the game of go without human knowledge," *Nature*, vol. 550, pp. 354–, Oct. 2017. [Online]. Available: http://dx.doi.org/10.1038/nature24270.

[59] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, *Wavenet: A generative model for raw audio*, 2016. arXiv: 1609.03499 [cs.SD].

[60] A. Holzinger, G. Langs, H. Denk, K. Zatloukal, and H. Müller, "Causability and explainability of artificial intelligence in medicine," *WIREs Data Mining and Knowledge Discovery*, vol. 9, no. 4, e1312, 2019. DOI: https://doi.org/10.1002/widm.1312. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/widm.1312. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/widm.1312.

[61] A. Mordvintsev, C. Olah, and M. Tyka, *Inceptionism: Going deeper into neural networks*, 2015. [Online]. Available: https://research.googleblog.com/2015/06/inceptionism-going-deeper-into-neural.html.

[62] M. T. Ribeiro, S. Singh, and C. Guestrin, *"Why should i trust you?: Explaining the predictions of any classifier*, 2016. arXiv: 1602.04938 [cs.LG].

[63] F. Doshi-Velez and B. Kim, *Towards a rigorous science of interpretable machine learning*, 2017. arXiv: 1702.08608 [stat.ML].

[64] Z. C. Lipton, *The mythos of model interpretability*, 2017. arXiv: 1606.03490 [cs.LG].

[65] G. Montavon, W. Samek, and K.-R. Müller, "Methods for interpreting and understanding deep neural networks.," *Digit. Signal Process.*, vol. 73, pp. 1–15, 2018. [Online]. Available: http://dblp.uni-trier.de/db/journals/dsp/dsp73.html#MontavonSM18.

[66] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, *Imagenet large scale visual recognition challenge*, 2015. arXiv: 1409.0575 [cs.CV].

[67] M. D. Zeiler, G. W. Taylor, and R. Fergus, "Adaptive deconvolutional networks for mid and high level feature learning," in *2011 International Conference on Computer Vision*, 2011, pp. 2018–2025. DOI: 10.1109/ICCV.2011.6126474.

[68] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller, *Striving for simplicity: The all convolutional net*, 2015. arXiv: 1412.6806 [cs.LG].

[69] A. Krizhevsky, G. Hinton, *et al.*, "Learning multiple layers of features from tiny images," 2009.

[70] J. Deng, W. Dong, R. Socher, L.-J. Li, Kai Li, and Li Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255. DOI: 10.1109/CVPR.2009.5206848.

[71] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, *Learning deep features for discriminative localization*, 2015. arXiv: 1512.04150 [cs.CV].

[72] S. Bazen and X. Joutard, "The taylor decomposition: A unified generalization of the oaxaca method to nonlinear models," 2013.

[73] G. Montavon, A. Binder, S. Lapuschkin, W. Samek, and K.-R. Müller, "Layer-wise relevance propagation: An overview," in *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, W. Samek, G. Montavon, A. Vedaldi, L. K. Hansen, and K.-R. Müller, Eds. Cham: Springer International Publishing, 2019, pp. 193–209, ISBN: 978-3-030-28954-6. DOI: 10.1007/978-3-030-28954-6_10. [Online]. Available: https://doi.org/10.1007/978-3-030-28954-6_10.

[74] M. Alber, S. Lapuschkin, P. Seegerer, M. Hägele, K. T. Schütt, G. Montavon, W. Samek, K.-R. Müller, S. Dähne, and P.-J. Kindermans, *Innvestigate neural networks!* 2018. arXiv: 1808.04260 [cs.LG].

[75] H. Jiang, J. Xu, R. Shi, K. Yang, D. Zhang, M. Gao, H. Ma, and W. Qian, "A multi-label deep learning model with interpretable grad-cam for diabetic retinopathy classification," in *2020 42nd Annual International Conference of the IEEE Engineering in Medicine Biology Society (EMBC)*, 2020, pp. 1560–1563. DOI: 10.1109/EMBC44109.2020.9175884.

[76] M. Böhle, F. Eitel, M. Weygandt, and K. Ritter, "Layer-wise relevance propagation for explaining deep neural network decisions in mri-based alzheimer's disease classification," *Frontiers in Aging Neuroscience*, vol. 11, 2019, ISSN: 1663-4365. DOI: 10.3389/fnagi.2019.00194. [Online]. Available: http://dx.doi.org/10.3389/fnagi.2019.00194.

[77] B. A. Toms, E. A. Barnes, and I. Ebert-Uphoff, "Physically interpretable neural networks for the geosciences: Applications to earth system variability," *Journal of Advances in Modeling Earth Systems*, vol. 12, no. 9, 2020, ISSN: 1942-2466. DOI: 10.1029/2019ms002002. [Online]. Available: http://dx.doi.org/10.1029/2019MS002002.

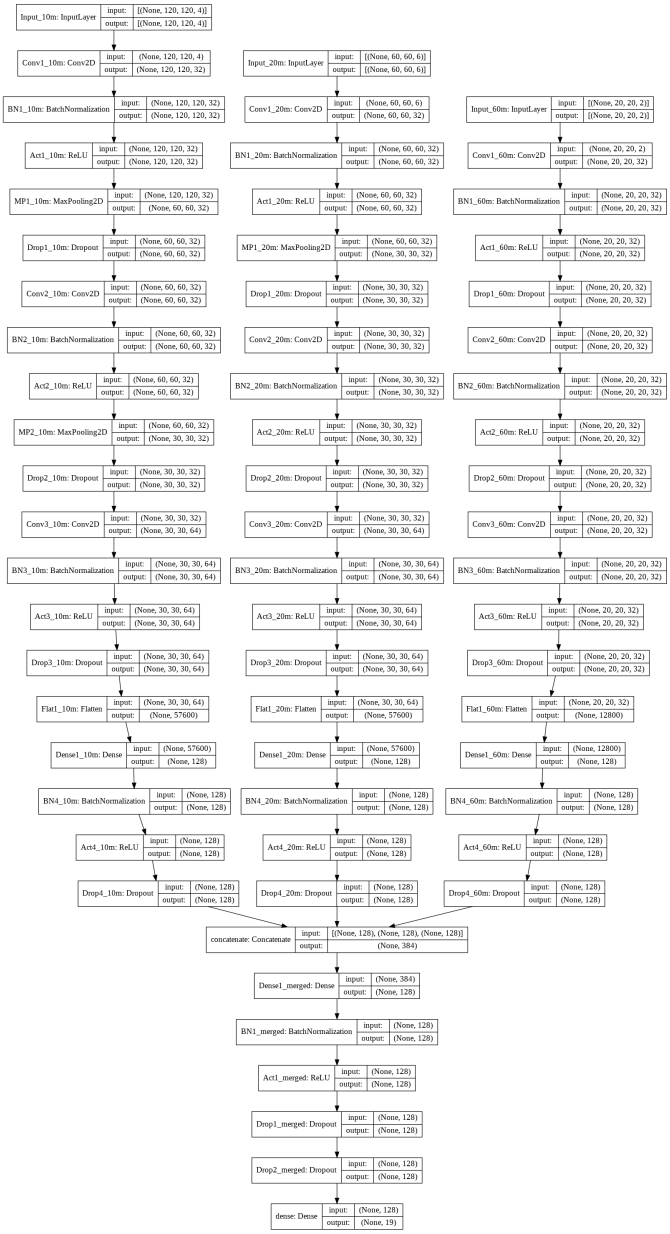
[78] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[79] F. Chollet, *Keras documentation: Computer vision*, 2020. [Online]. Available: https://keras.io/examples/vision/.

[80] M. S. Sorower, "A literature survey on algorithms for multi-label learning," 2010.

Appendix

A.1 First Appendix



K-branch reference model architecture

A.2 Second Appendix

Model alias	Micro-average F1-score
2DConv_base	0.44714
10mConv1Dil2	0.45331
10mConv2Dil2	0.45379
10mConv3Dil2	0.44645
10mConvAllDil2	0.41138
20mConv2Dil2	0.45758
20mConv3Dil2	0.42049
20mConvAllDil2	0.43116
60mConv2Dil2	0.44186
60mConv3Dil2	0.45398
60mConvAllDil2	0.47254
3DConv_base	0.49411
3D_10mConv1Dil2	0.45784
3D_10mConv2Dil2	0.49087
3D_10mConv3Dil2	0.48516
3D_10mConvAllDil2	0.4779
3D_20mConv1Dil2	0.48168
3D_20mConv2Dil2	0.48342
3D_20mConv3Dil2	0.4749
3D_20mConvAllDil2	0.4743
3D_60mConv1Dil2	0.4554
3D_60mConv2Dil2	0.48538
3D_60mConv3Dil2	0.47972
3D_60mConvAllDil2	0.45807

Micro average F1-score for experimental models