

SHIP DETECTION USING SATELLITE IMAGERY MACHINE LEARNING MODEL

A Project Report

Submitted in the partial fulfillment of the Requirements for the award of Degree of

BACHELOR OF TECHNOLOGY IN INFORMATION TECHNOLOGY

SUBMITTED BY:

SHASHANK DIWAKAR (50674)

AKSHAY NAGPAL (50612)

ANKITA SATYARTHI (50645)

SARTHAK MITTAL (50261)

Under the able guidance of

Er. SUBODH PRASAD, Assistant Professor, ITD



**DEPARTMENT OF INFORMATION TECHNOLOGY
COLLEGE OF TECHNOLOGY
GOVIND BALLABH PANT UNIVERSITY OF AGRICULTURE AND
TECHNOLOGY, PANTNAGAR-263145
DISTRICT- UDHAM SINGH NAGAR (UTTARAKHAND)**

July, 2020



CERTIFICATE

This is to certify that project entitled “SHIP DETECTION USING SATELLITE IMAGERY MACHINE LEARNING MODEL” which is submitted by:

<u>Group Members</u>	<u>Id. No.</u>
SHASHANK DIWAKAR	50674
AKSHAY NAGPAL	50612
SARTHAK MITTAL	50261
ANKITA SATYARTHI	50645

This project is a record of student's work carried out by them in the partial fulfillment of the requirements for the award of degree of Bachelor of Technology in Information Technology, College of Technology, Govind Ballabh Pant University of Agriculture & Technology, Pantnagar.

Er. Subodh Prasad
Asstt. Professor
Project Guide

Prof. (Dr.) H.L. Mandoria
Professor & Head
Dept. of Information Technology

**DEPARTMENT OF INFORMATION TECHNOLOGY
COLLEGE OF TECHNOLOGY
GOVIND BALLABH PANT UNIVERSITY OF AGRICULTURE AND
TECHNOLOGY, PANTNAGAR-263145
DISTRICT- UDHAM SINGH NAGAR (UTTARAKHAND)**

ACKNOWLEDGEMENT

The completion of this project is not the beginning of an end but the beginning of a learning experience, the value of which cannot be measured quantitatively. The efforts put up by us during the development of this report would not have been fruitful in the absence of those people around us who encouraged us all the time.

We are highly indebted to our **Head of Department, Prof. (Dr.) H.L. Mandoria** for providing us a platform to learn and excel. We are also thankful to the laboratory staff who has cooperated us as and when required to access the facilities of laboratories in the department. Finally, we express our deepest gratitude towards **Er. Subodh Prasad, Assistant Professor, Department of Information Technology**, our mentor during the course of this project, for his valuable guidance, precious time and kind gestures during the development of this project. His knowledge and experience proved to be of utmost value. We appreciate his guidance throughout.

We gratefully acknowledge all the faculty members **Dr. Ratnesh kumar, Dr. Shri Prakash Diwedi, Er. Ashok Kumar, Er. Sanjay Joshi, Er. B.K. Pandey, Er. Govind Verma, Er. Rajesh Shyam Singh, Er. Suchit Gupta, Er. Shikha Goswami** of Information Technology, College of Technology, Govind Ballabh Pant University of Agriculture and Technology, Pantnagar, for their constant support and encouragement during the project.

GROUP MEMBERS

ID NO.

SIGNATURE

SHASHANK DIWAKAR

50674

AKSHAY NAGPAL

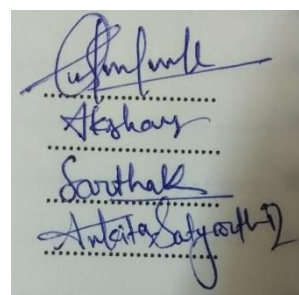
50612

SARTHAK MITTAL

50261

ANKITA SATYARTHI

50645



LIST OF FIGURES

4.1	IMAGES LABELLED AS SHIP.	20
4.2	IMAGES LABELLED AS NO-SHIP.	20
4.3	ZERO PADDING ADDED TO AN IMAGE.	22
4.4	FILTER APPLICATIONS ON AN INPUT MATRIX.	23
4.5	MAX POOLING EXAMPLE	24
4.6	LAYER ARCHITECTURE OF MODEL.	24
4.7	FLATTENING OF A POOLED FEATURE.	25
4.8	OPERATION FLOW DIAGRAM.	27
5.1	DIMENSION CONSIDERATION WHILE SCANNING FOR SHIP IN A SATELLITE IMAGE	29
5.2	INPUT GIVEN TO ML MODEL.	30
5.3	FIRST PREDICTED SHIP.	31
5.4	SECOND PREDICTED SHIP.	32
5.5	THIRD PREDICTED SHIP.	33
5.6	FOURTH PREDICTED SHIP.	34
5.7	FIFTH PREDICTED SHIP.	35
5.8	SIXTH PREDICTED SHIP.	36
5.9	SEVENTH PREDICTED SHIP.	37
5.10	EIGHT PREDICTED SHIP.	38
5.11	NINTH PREDICTED SHIP.	39
5.12	TENTH PREDICTED SHIP.	40
5.13	ELEVENTH PREDICTED SHIP.	41
5.14	TWELFTH PREDICTED SHIP.	42
5.15	FINAL RESULT WITH ALL MARKED SHIPS	43

ABBREVIATIONS

- **SAR** Synthetic Aperture Radar
- **VHF** Very High Frequency
- **AIS** Automated Identification System
- **ML** Machine Learning
- **NFR** Non Functional Requirement
- **SIANN** Space Invariant Artificial Neural Network
- **CNN** Convolution Neural Network
- **SGD** Stochastic Gradient Descent
- **RBG** Red Blue Green
- **ReLU** Rectified Linear Unit

CONTENTS

ACKNOWLEDGEMENT	iii
LIST OF FIGURES	iv
ABBREVIATIONS	v
ABSTRACT	vii
Chapter 1: Introduction	
1.1 Existing System	9
1.2 Introduction to the system	9
Chapter 2: Review of Literature	
2.1 Literature	11
2.2 Proposed System	11
2.3 Benefits of proposed model	12
Chapter 3: Problem Specification	
3.1 Existing Problem	14
3.2 Objective of the Project	14
3.3 General Requirements	14
Chapter 4: Materials and Methodology	
4.1 Specific Requirements	18
4.2 Implementation	18
Chapter 5: Results and Discussion	29
Chapter 6: Conclusion	
6.1 Conclusion	45
6.2 Future scope of work	45
References	47
Appendix A: Project Source Code	49
Appendix B: Abstract in Hindi	56
Appendix C: Vitae	59

ABSTRACT

Ship detection across the wide spread sea is a tedious job which was done manually in the previous years. Before the digital age, the scheduling of port activities like movement of cargo and loading and unloading of ships and custom activities had to be done manually which was a tedious and time taking job. So, there was a need to detect these coming ships as early as possible so that the port manager could schedule the ports activities accordingly to increase the efficiency and many technologies were developed in this process to establish the connection between the ships and the port. Then with the advancement of technology various methods were developed for ships monitoring.

SAR (Synthetic Aperture Radar) is still the leading technology for maritime monitoring. Vessel identification and communication is done typically through the use of Automated Identification System (AIS), which uses VHF radio frequencies to wirelessly broadcast the ship location, destination and identify to nearby receiver devices on other ships and land-based systems.

Sometimes due to bad weather, some ships get lost in the wide spread sea due to the lost connection or due to any other technological failure. At that time locating these ships becomes a tedious job and not easy at all and we have to scan a large portion of sea with the help of satellite images. The flood of new imagery is outgrowing the ability for organizations to manually look at each image that gets captured, and there is a need for machine learning and computer vision algorithms to help automate the analysis process, and our Ship Detection Model provides this automation. Ship Detection Using Satellite Imagery is a Machine Learning model aimed at detection of ships around the sea ports using live satellite imagery. The aim of this model is to help address the difficult task of detecting the location of large ships in satellite images.

Further, automating this process can be applied to many issues including monitoring port activity levels and supply chain analysis, by applying our ship detection model we will be able to identify any incoming ships using live satellite imagery so that the port activities can be scheduled accordingly.



INTRODUCTION

1.1 Existing System:

Presently, tracking of ships near the ports is done manually or through software using SAR technology creating two-dimensional images or three-dimensional reconstruction of objects using the motion of the radar antenna over a target region and worldwide used VHF radio technology of two way radio transceivers on ships for bidirectional voice communication from ship-to-ship or ship-to-shore.

There is a constant need of continuous monitoring of ships and activities all around the port. Precisely detecting a floating vessel across the vast sea can be a challenging task because of increasing pollution and many other impurities such as a wooden log etc. which can mistakenly to be a small size boat. So, there is a need to detect the ships as early and accurately as possible so that any loss can be minimized.

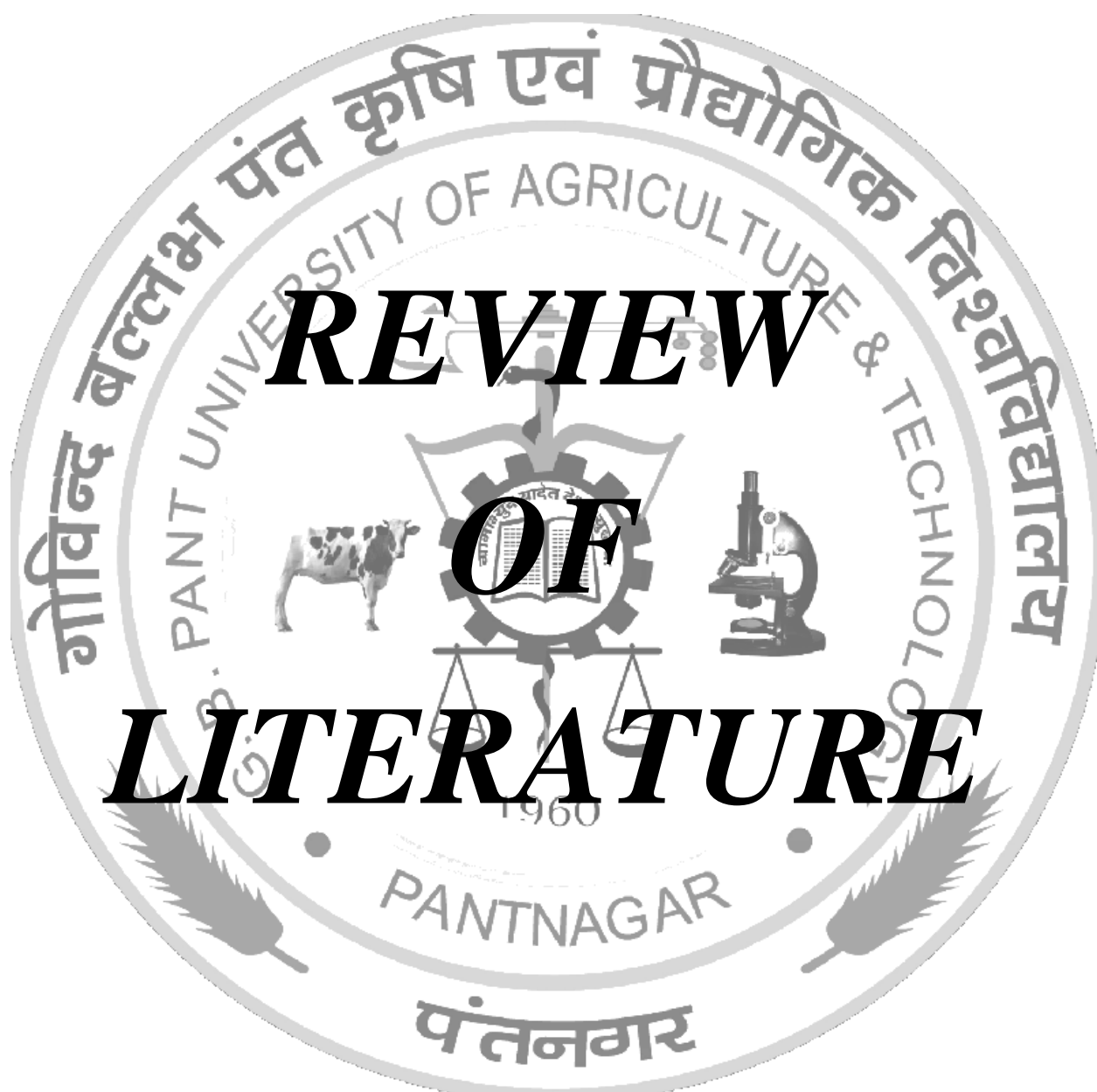
1.2 Introduction to the system:

Scanning a wide spread sea to detect a ship using SAR technology is impractical and due to this we focus ourselves on the satellite images by the means of which we can scan a wider range of sea in a small time. The project aims to bring automation in the ship detection process using Machine Learning to detect any ships in the given radius around the port. This project covers the complete pipeline from the raw dataset to the deployable machine learning model. This is proposed as an artificial intelligence based solution to the ever-growing challenge of applying the high degree implementation of ship detection and identification.

Through automating this process of applying machine learning end to end additionally offers the advantages of producing simpler solution, faster creation of this solution. The word ‘Detection’ is used in the literature sometimes in a wide and sometimes in a narrow sense. The complete vessel detection procedure refers to the detection in wide sense, and is completed in three main sequential steps.

1. Ship Detection
2. Ship Classification
3. Ship Identification

In our model, our main focus is limited to the task of ship detection only. This model is developed in PYTHON and JSON using Keras (an open-source neural-network library written in Python) and Matplotlib (a plotting library for the python and its numerical extension NumPy. We train our model with satellite images of a fixed file format and dimension. After the successful training of this model this ML model can determine whether any given satellite image contain a vessel or not.



REVIEW OF LITERATURE

2.1 Literature:

Although SAR (Synthetic Aperture Radar) is still the leading technology for maritime monitoring, the number of studies based on optical satellite data is quickly growing. We start by introducing all the existing sensor systems for vessel detection, but subsequently focus only on optical imaging satellites. Various recent studies of the temporal development of optical satellite characteristics connect this to vessel detection. The methods used for optical imagery-based vessel detection and classification in detail have achieved detection accuracies, and there is a possibilities for fusing optical data with other data sources. The most common factors greatly influencing the vessel detection accuracy are the following: different weather conditions affecting sea surface characteristics, the quantity of clouds and haze, solar angle, and imaging sensor characteristics. All these factors bring great variations in the selection of the most suitable method; some still continue to pose unsolved challenges. For higher relevance and wider usage, we suggest that the algorithms for detection and classification should support a variety of targets and meteorological conditions, and ideally also a variety of optical satellite sensors. At least, they should be tested on many images under different conditions. This is not usually the case in the existent literature. It can be seen that vessel monitoring from space borne optical images is a popular topic and has a great operational potential in the near future due to the large amount of satellite data, much of it free and open.

2.2 Proposed Model:

The project aims to bring automation in the ship detection process using Machine Learning to detect any ships in the given radius around the port. This automated project covers the complete pipeline from the raw dataset to the deployable machine learning model. This is proposed as an artificial intelligence based solution to the ever-growing challenge of applying the high degree implementation of ship detection and identification. We first train our model with around 4000 satellite images of fixed dimensions (80 x 80 RGB) pre-classified as “ship” and “No-ship” and then provide some testing data on which the algorithm runs and give us the result as “ship” or “no-ship”. Training of model takes around one and a half hour on a core i5 processor.

2.3 Benefits of Project:

- Automation in ship detection
- Cost Efficient
- Computational Efficient
- Maritime Security



PROBLEM SPECIFICATION

3.1 Existing problem

Ship detection from remote sensing is a crucial application for maritime security which includes among other traffic surveillance, protection against illegal fisheries, oil discharge control and sea pollution monitoring. This is done typically through the use of Automated Identification System (AIS), which uses VHF radio frequencies to wirelessly broadcast the ship location, destination and identify to nearby receiver devices on other ships and land-based systems. AIS are very effective at monitoring ships which are legally required to install a VHF transponder, but fail to detect those which disconnect their transponder.

Similarly when a ship get lost in the sea we cannot use SAR to produce 3D images of each and every object we came across while scanning for our lost vessel in the wide sea because of so many impurities present in the sea.

This is where satellite imagery can help. By analyzing the real time images through satellite, our model will scan the larger portion of sea in a less time and detect all the ships no matter they are connected to a transponder or not.

3.2 Objective of the project

The purpose of this project is to develop a ML model working on satellite images which will efficiently help various private and government agencies to uniquely identify the ships in the oceans and help regulate them to control and monitor various sea and port activities. The aim of this model is to accurately identify a ship or vessel and inform the authorities about their presence as well as to find any lost vessel or ship. This model can be effectively used in defence services, managing port activities or in managing sea traffic control. The benefit of this model is this model can also be used in automating the manual process including monitoring port activity levels and supply chain analysis.

3.3 General Requirements:

- **Language:** Python, JSON, NumPy
- **Programs & Library:** KERAS, Matplotlib, OpenCV
- **Dataset:** Planet's Open California dataset, shipsnet.zip

3.3.1 Functional and Non-Functional Requirements

The requirements for developing any machine learning model can be broadly classified into two categories:

- Functional requirements
- Non Functional requirements

Functional Requirements

In ML systems, the quality of the resulting predictions can be considered a functional requirement. Some of these are:

1. **Elicitation:** In the elicitation process, we should identify “protected” characteristics that must not be used by the ML algorithm to discriminate test samples. We should also identify all possibly relevant sources of data that may help the ML system to provide good and robust results.

Ex: Satellite imagery used to build this dataset is made available through Planet's Open California dataset, which is openly licensed. These images are taken from zipped directory shipsnet.zip that contains the entire dataset as .png image chips.

2. **Analysis:** Most important in requirements analysis for ML systems are to define and discuss the performance measures and expectations by which the ML system shall be assessed. Performance measures such as accuracy, precision, or recall. It is also important to analyse whether false positives and false negatives are equally bad. An early step in ML development is to understand the quality of the data (e.g., completeness, sparseness, and consistency) and how it must be enriched to constitute feasible training data.

Ex: The performance efficiency of our model is 85-90%.

3. **Specification:** Identify and specify requirements regarding the collection of data, the data formats, and the ranges of data. Specifying data requirements includes information about the necessary quantity and quality of data. The requirements specification should also contain statements about the expected predictive power expressed in terms of the performance measures discussed during requirements elicitation and analysis.

Ex: Our data set include 4000 80x80 RGB images & are orthorectified to a 3meter pixel size. . Each individual image filename follows a specific format: {label} __ {scene id} __ {longitude} _ {latitude}.png

4. **Verification & Validation:** Due to the dependency between the behaviour of an ML system and the data it has been trained on, it is crucial to define actions that ensure that training data actually corresponds to real data. Since data characteristics in reality may change over time, requirements validation becomes an activity that needs to be performed continuously during system operation. We should also specify conditions for data anomalies that may potentially lead to unreasonable behaviour of the ML system during

runtime. Apart from runtime monitoring, requirements validation also includes analysing the training and production data for bias and imbalances.

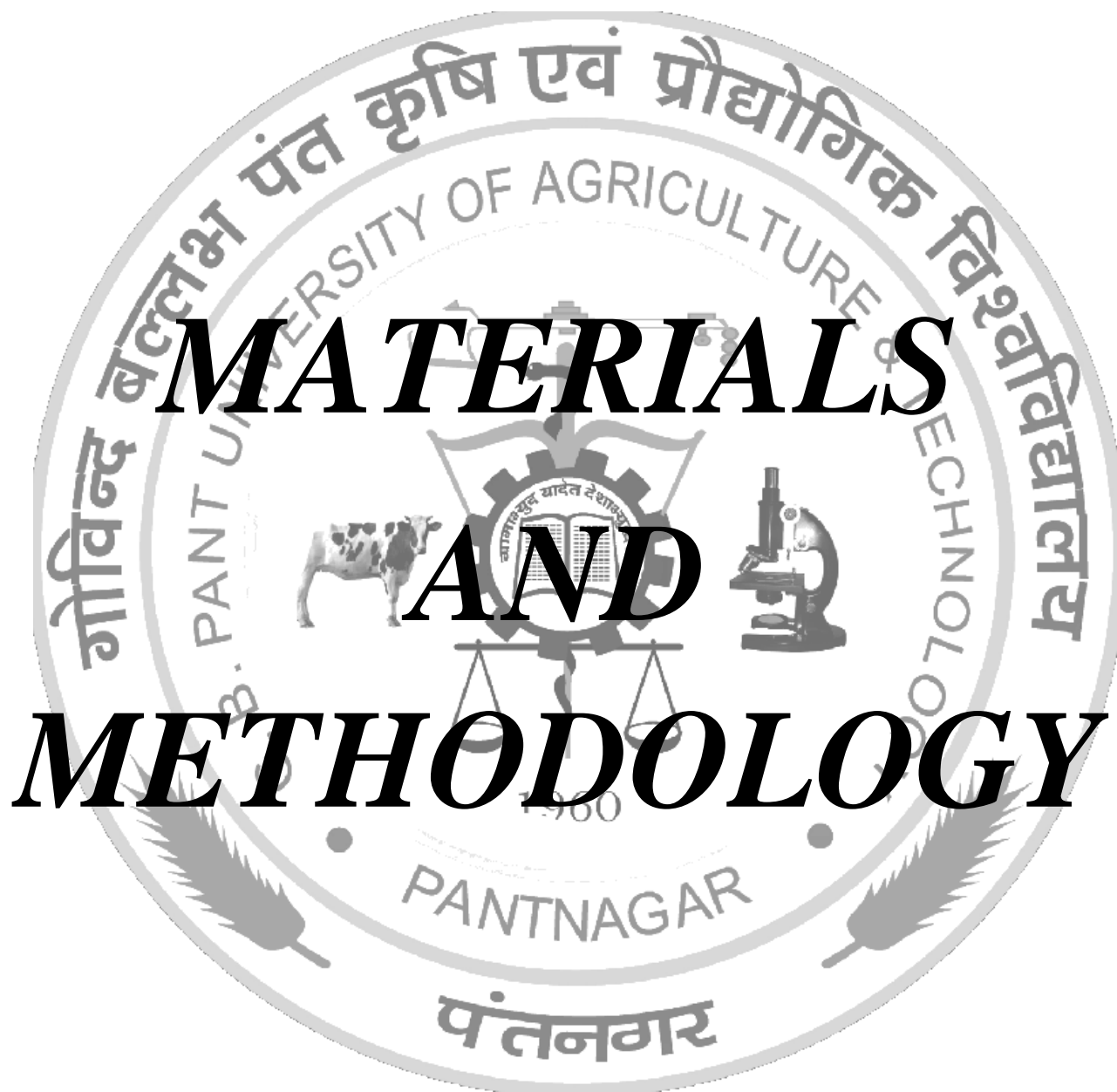
Ex: Images with bright pixels can cause false results.

Non Functional Requirements

Machine Learning (ML) provides approaches which use big data to enable algorithms to “learn”, producing outputs which would be difficult to obtain otherwise. Despite the advances allowed by ML, much recent attention has been paid to certain qualities of ML solutions, particularly fairness and transparency, but also qualities such as privacy, security, and testability.

1. **Accuracy & Performance:** Most ML work reports on algorithm accuracy (often precision and recall), i.e., how “correct” the output is compared to reality.
2. **Fairness:** Recent work has focused on technical solutions to make ML algorithms more fair, finding that the removal of sensitive features (e.g., race, gender) is not sufficient to ensure fair results, and considering the trade-off between fairness and other NFRs. Work in this area has attempted to find mathematical or formal definitions of fairness, e.g. statistical parity, individual fairness, and has found that the accurate implementation of fairness depends more on how fairness is defined and measured than how it is implemented.
3. **Security & Privacy:** Efforts have been made to address privacy concerns when using big (often personal) data to facilitate ML. Introduces protocols for preserving privacy in various ML approaches, and explicitly acknowledges the trade-off in terms of algorithm speed when revising techniques for privacy.
4. **Testability:** Work exists which considers systematically testing the outcome of ML systems.
5. **Reliability:** Further work has considered reliability in ML, e.g., looking at the reliability of individual ML predictions, focusing on reliability estimation.

Thus, it is critical to capture these requirements as accurately and exhaustively as possible and ensure that the machine learning model design addresses each requirement.



MATERIALS AND METHODOLOGY

4.1 Specific Requirements:

4.1.1 Software and Languages used:

- Software used to run this model is JUPITER.
- The programming language used for its development is PYTHON, JSON and NumPy.
- Open source neural network library used is Keras.
- Plotting library used for the Python and its numerical extension is Matplotlib.
- Open source computer vision library used is OpenCV.
- Dataset used to train the model is Planet's Open California dataset, shipsnet.zip

4.1.2 Hardware Used:

- Core-i5 64-bit processor.
- 8GB of memory.
- Nvidia GTX 1080.

4.2 Implementation

4.2.1 Approach to solve the problem:

Building a Machine learning begins with identifying the kind of ML model to build upon. Since the project is based on image processing and identifying the specific features in the images to conclude the existence of ship in satellite imagery.

Convolutional neural network: In deep learning, a convolution neural network (CNN, or ConvNet) is a class of deep neural networks, most commonly applied to analyzing visual

imagery. They are also known as shift invariant or space invariant artificial neural networks (SIANN), based on their shared-weights architecture and translation invariance characteristics. They have applications in image and video recognition, recommender systems, image classification, medical image analysis, natural language processing, and financial time series.

CNNs are regularized versions of multilayer perceptron. Multilayer perceptron usually mean fully connected networks, that is, each neuron in one layer is connected to all neurons in the next layer. A convolutional neural network consists of an input and an output layer, as well as multiple hidden layers. The hidden layers of a CNN typically consist of a series of convolutional layers that convolve with a multiplication or other dot product. The activation function is commonly a ReLU layer, and is subsequently followed by additional convolutions such as pooling layers, fully connected layers and normalization layers, referred to as hidden layers because their inputs and outputs are masked by the activation function and final convolution.

We have used CNN because pre-processing required in CNN is much lower as compared to other classification algorithm. In other methods Filters are hand-engineered while in CNN it has the ability to learn these filters/characteristics. A ConvNet is able to successfully capture the spatial & temporal dependencies in an image through the application of relevant filters.

Input Image: We have RGB images. The role of ConvNet is to reduce the image into a form which is easier to process, without losing features which are critical for good predictions.

Each individual image filename follows a specific format: {label} __ {scene id} __ {longitude} _ {latitude}.png

- label: Valued 1 or 0, representing the "ship" class and "no-ship" class, respectively.
- scene id: The unique identifier of the PlanetScope visual scene the image chip was extracted from. The scene id can be used with the Planet API to discover and download the entire scene.
- longitude_latitude: The longitude and latitude coordinates of the image center point, with values separated by a single underscore.

The pixel value data for each 80x80 RGB image is stored as a list of 19200 integers within of the data list. The first 6400 entries contain the red channel values, the next 6400 the green, and the final 6400 the blue. The image is stored in row-major order, so that the first 80 entries of the array are the red channel values of the first row of the image.

The list values at index i in labels, scene_ids, and locations each correspond to the i -th image in the data list.

Following are the samples of images which are used to train the Machine Learning Model.

The "ship" class includes 1000 images. Images in this class are near-centered on the body of a single ship. Ships of different ship sizes, orientations, and atmospheric collection conditions are included. Example images from this class are shown below.



FIG: 4.1 IMAGES CLASS LABELLED AS “SHIP”

The "no-ship" class includes 3000 images. A third of these are a random sampling of different land cover features - water, vegetation, bare earth, buildings, etc. - that do not include any portion of a ship. The next third are "partial ships" that contain only a portion of a ship, but not enough to meet the full definition of the "ship" class. The last third are images that have previously been mislabeled by machine learning models, typically caused by bright pixels or string linear features. Example images from this class are shown below

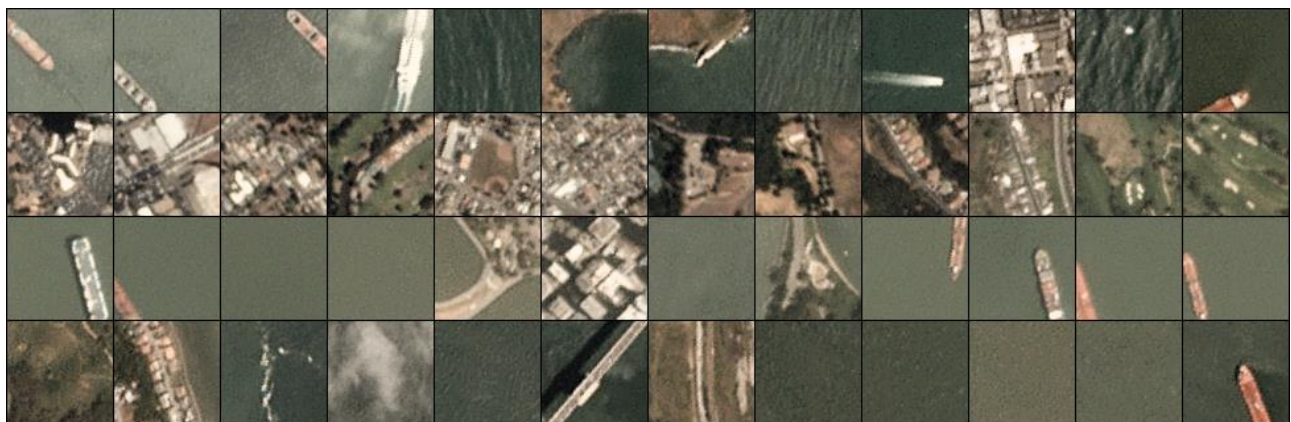


FIG: 4.2 IMAGES CLASS LABELLED AS “NO-SHIP”

Convolutional Layers: A convolutional layer contains a set of filters whose parameters need to be learned. The height and weight of the filters are smaller than those of the input volume. Each filter is convolved with the input volume to compute an activation map made of neurons. In other words, the filter is slid across the width and height of the input and the dot products between the input and filter are computed at every spatial position. The output volume of the convolutional layer is obtained by stacking the activation maps of all filters along the depth dimension. Since the width and height of each filter is designed to be smaller than the input, each neuron in the activation map is only connected to a small local region of the input volume. In other words, the receptive field size of each neuron is small, and is equal to the filter size. The local connectivity is motivated by the architecture of the animal visual cortex where the receptive fields of the cells are small. The local connectivity of the convolutional layer allows the network to learn filters which maximally respond to a local region of the input, thus exploiting the spatial local correlation of the input (for an input image, a pixel is more correlated to the nearby pixels than to the distant pixels). In addition, as the activation map is obtained by performing convolution between the filter and the input, the filter parameters are shared for all local positions. The weight sharing reduces the number of parameters for efficiency of expression, efficiency of learning, and good generalization.

Convolution Operation: Convolution is a linear operation that involves the multiplication of a set of weights with the input, much like a traditional neural network. Given that the technique was designed for two-dimensional input, the multiplication is performed between an array of input data and a two-dimensional array of weights, called a filter or a kernel.

In RGB images, Kernel/filter has the same depth as that of input image. Matrix multiplication is performed & all the results are summed with the bias to give us a squashed one-depth channel Convolved Feature Output. Objective of the convolution operation is to extract the high-level features such as edges, color, gradient operation etc.

Two types of convolution operation are:

- Convolved feature is reduced in dimensionality as compared to the input image.
- Convolved feature dimensionality is either increased or remains the same as original.

Padding: Padding is the operation to increase the size of the input image. Padding is simply a process of adding layers of zeros to our input images so as to avoid the problems mentioned above.

Two types of padding are:

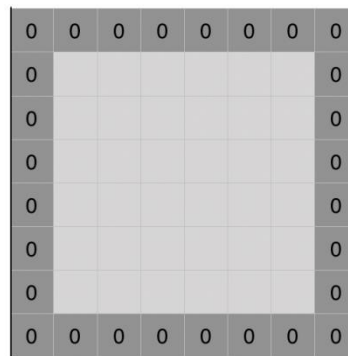
- **Valid Padding:** It implies no padding at all. The input image is left in its valid/unaltered shape.

$$[(n \times n) \text{ image}] * [(f \times f) \text{ filter}] \longrightarrow [(n - f + 1) \times (n - f + 1) \text{ image}]$$

- **Same Padding:** In this case, we add ‘p’ padding layers such that the output image has the same dimension as the input image.

$$[(n + 2p) \times (n + 2p) \text{ image}] * [(f \times f) \text{ filter}] \longrightarrow [(n \times n) \text{ image}]$$

where * represents a convolution operation.



Zero-padding added to image

FIG: 4.3 ZERO PADDING ADDED TO AN IMAGE

Filter: The filter is smaller than the input data and the type of multiplication applied between a filter-sized patch of the input and the filter is a dot product. A dot product is the element-wise multiplication between the filter-sized patch of the input and filter, which is then summed, always resulting in a single value. Because it results in a single value, the operation is often referred to as the “scalar product”.

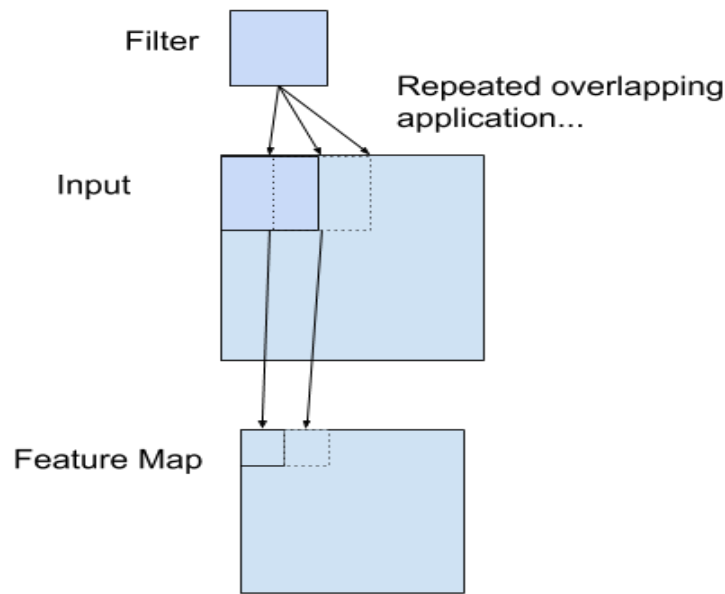


FIG: 4.4 FILTER APPLICATIONS ON AN INPUT MATRIX

Using a filter smaller than the input is intentional as it allows the same filter (set of weights) to be multiplied by the input array multiple times at different points on the input. Specifically, the filter is applied systematically to each overlapping part or filter-sized patch of the input data, left to right, top to bottom.

Once a feature map is created, we can pass each value in the feature map through a non-linearity, such as a ReLU, much like we do for the outputs of a fully connected layer.

The innovation of using the convolution operation in a neural network is that the values of the filter are weights to be learned during the training of the network.

The network will learn what types of features to extract from the input. Specifically, training under stochastic gradient descent, the network is forced to learn to extract features from the image that minimize the loss for the specific task the network is being trained to solve i.e. Identifying the features thoroughly to conclude the existence of ship.

Pooling Operation: The pooling layers operate independently on every depth slice of the input and resizes it spatially. The most common form is a pooling layer with filters of size 2×2 applied with a stride of 2 down samples at every depth slice in the input by 2 along both width and height, discarding 75% of the activations.

Pooling type: Max Pooling

Pool size: 2×2

Example:

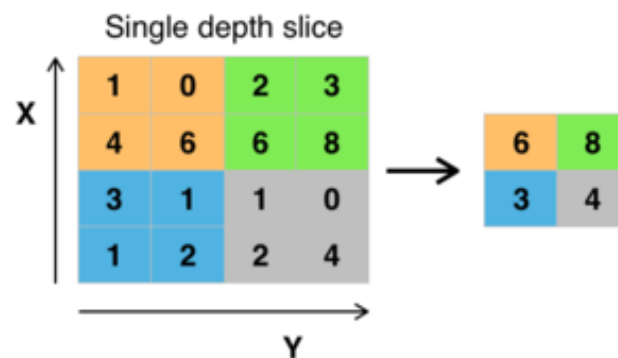


FIG: 4.5 MAX POOLING EXAMPLE

Layers Architecture: In this model, the convolution and pooling operation are used in round robin fashion.

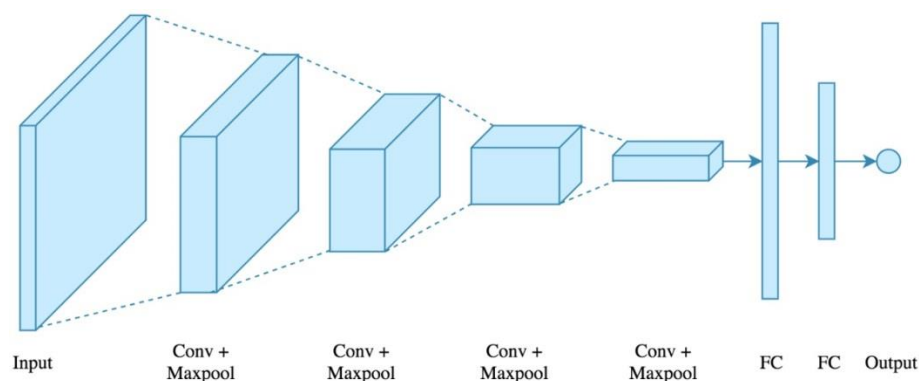


FIG: 4.6 LAYER ARCHITECTURE OF MODEL

We have the input of 80*80 images which after a single convolution layer and a Max Pooling layer becomes 40*40 size input images to next layer. Similarly, after applying total of 4 rounds of convolution layer and max pooling layer we are left with 5*5 size image on which we apply flatten operation.

Flatten operation tend the image matrix to align in a single array.

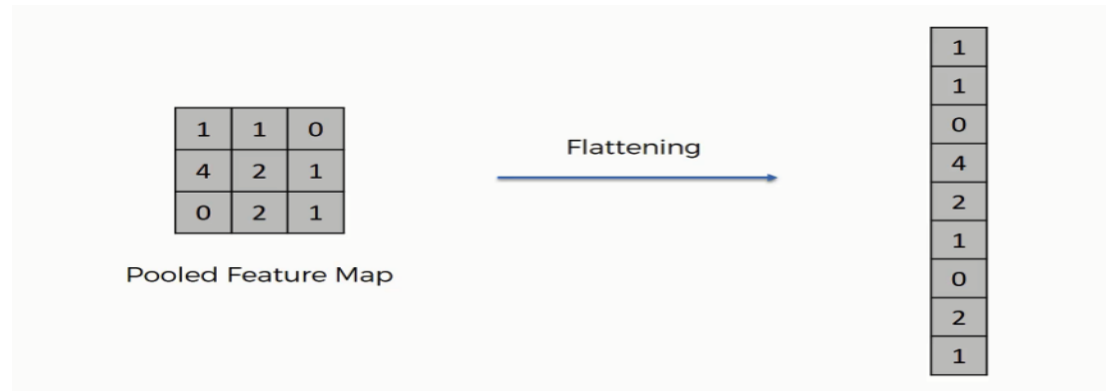


FIG: 4.7 FLATTENING OF A POOLED FEATURE

Over flattening operation, softmax layer is applied which gives us a discrete probability distribution over the two classes in our problem of binary classification.

Softmax Function: Softmax function also known as softargmax or normalized exponential function is a function that takes as input a vector z of K real numbers, and normalizes it into a probability distribution consisting of K probabilities proportional to the exponentials of the input numbers. That is, prior to applying softmax, some vector components could be negative, or greater than one; and might not sum to 1; but after applying softmax, each component will be in the interval $(0,1)$, and the components will add up to 1, so that they can be interpreted as probabilities.

That is the individual probabilities p_i are between 0 and 1, and the total probability $\sum_{i=1}^N p_i = 1$, so we have a probability measure.

Having discrete probability value of each image matrix allow us to compare with the actual value label so we can decrease the error in our model by applying stochastic gradient descent algorithm in backpropagation stage of neural network.

Stochastic Gradient Descent (SGD): In a Stochastic Gradient Descent, a few samples are selected randomly instead of the whole data set for each iteration. In Gradient Descent, there is a term called “batch” which denotes the total number of samples from a dataset that is used for calculating the gradient for each iteration. In typical Gradient Descent optimization, like Batch Gradient Descent, the batch is taken to be the whole dataset. Although, using the whole dataset is really useful for getting to the minima in a less noisy and less random manner, but the problem arises when our datasets gets big. In SGD, it uses only a single sample, i.e., a batch size of one, to perform each iteration. The sample is randomly shuffled and selected for performing the iteration.

4.2.2 Implementation of the project:

In our model, we have used:

- Filter/Kernel used: 3*3
- Activation function used: ReLU
- Convolution Dimension: 2D
- Padding: Same
- Pooling type: MaxPooling
- Pool size: 2*2

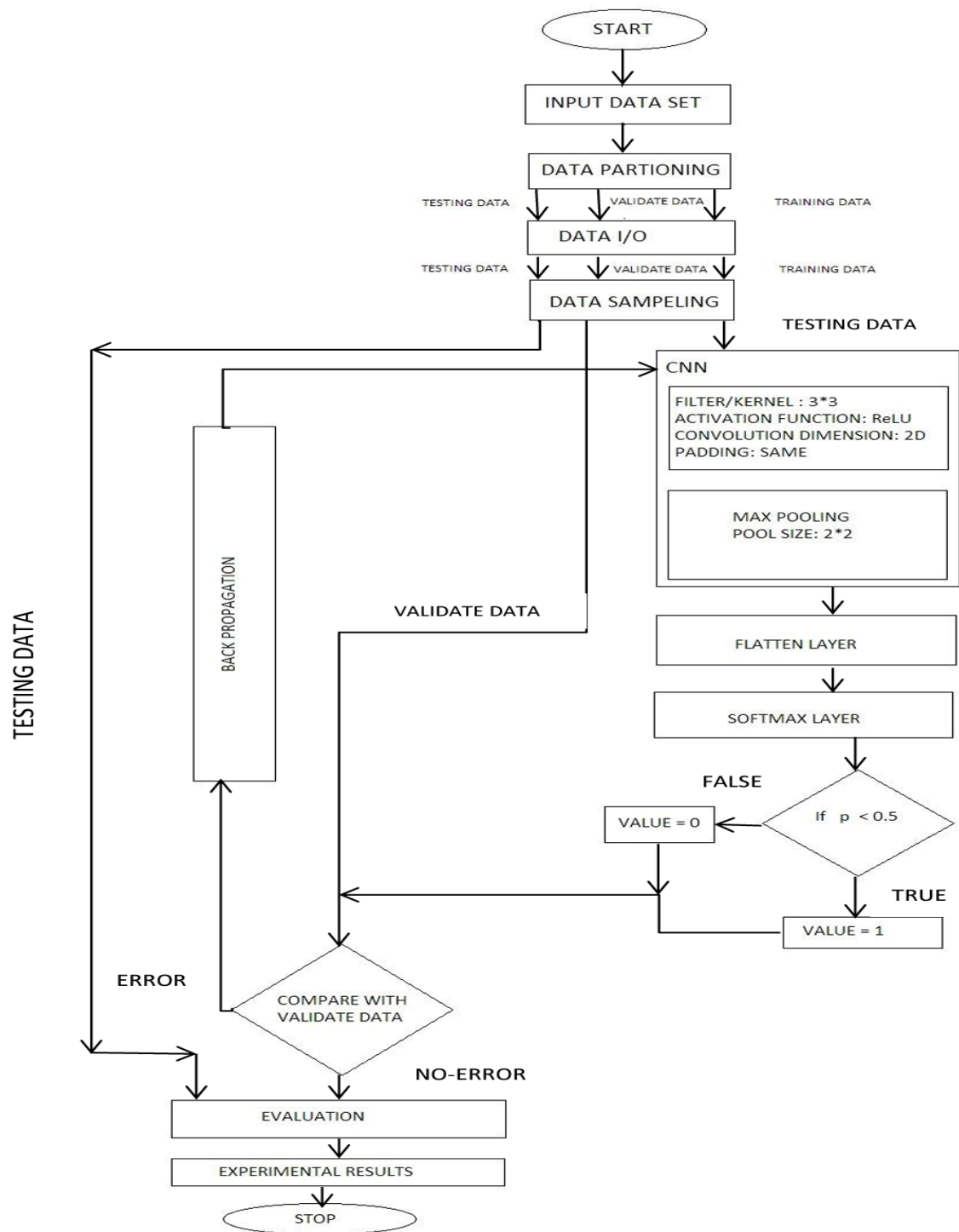


FIG: 4.8 OPERATIONAL FLOW DIAGRAM



RESULTS AND DISCUSSION

When we start working with our ML model the first thing that we need to do is to define how a ship might look in an image taken from a satellite.

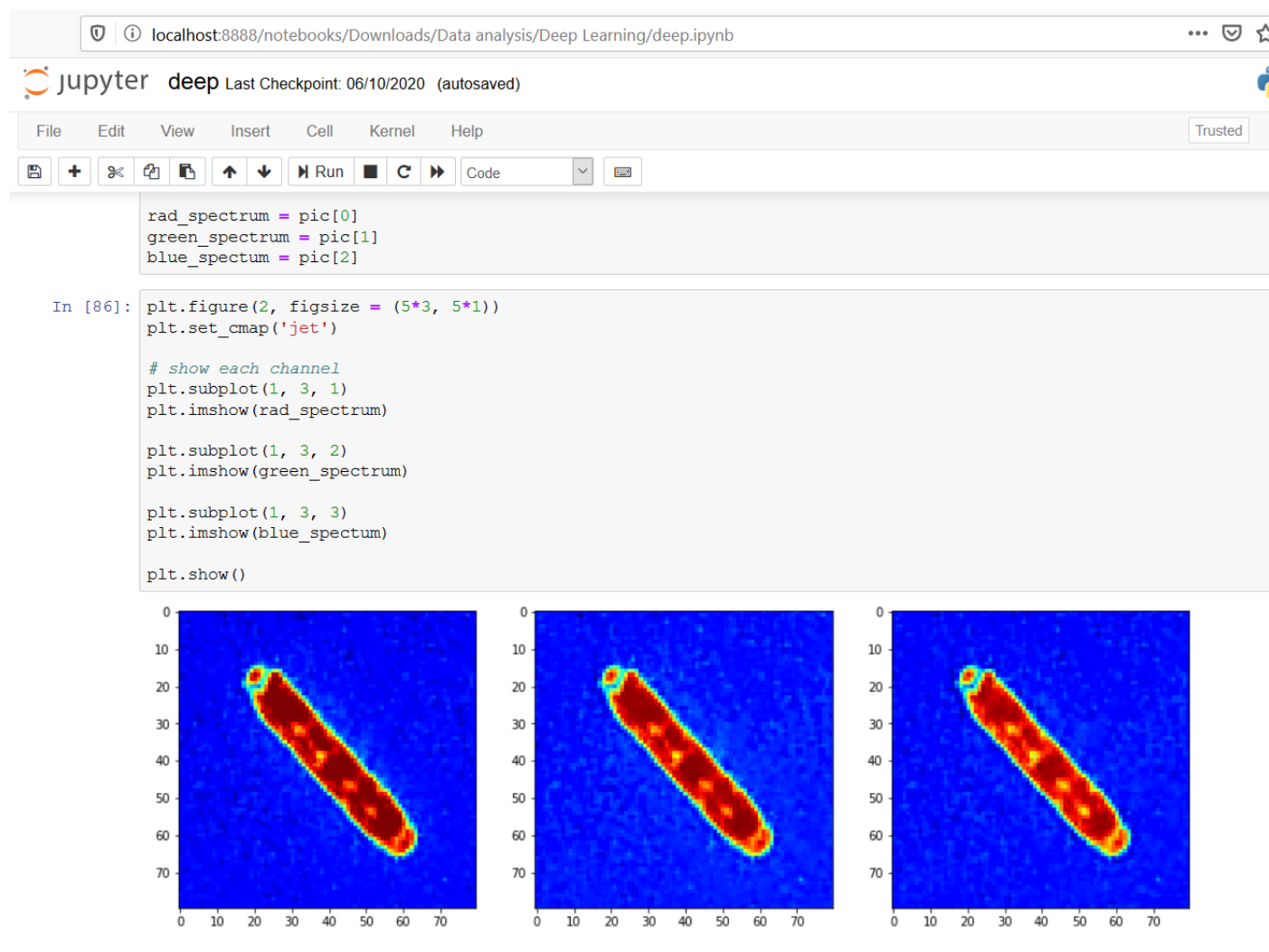


FIG: 5.1 DIMENSION CONSIDERATIONS WHILE SCANNING SHIPS

Figure 5.1 shows the dimensions and the shape that the Deep Learning model needs to consider while scanning the satellite image for any incoming ship.

Figure 5.2 shows the input satellite image which is to be processed.

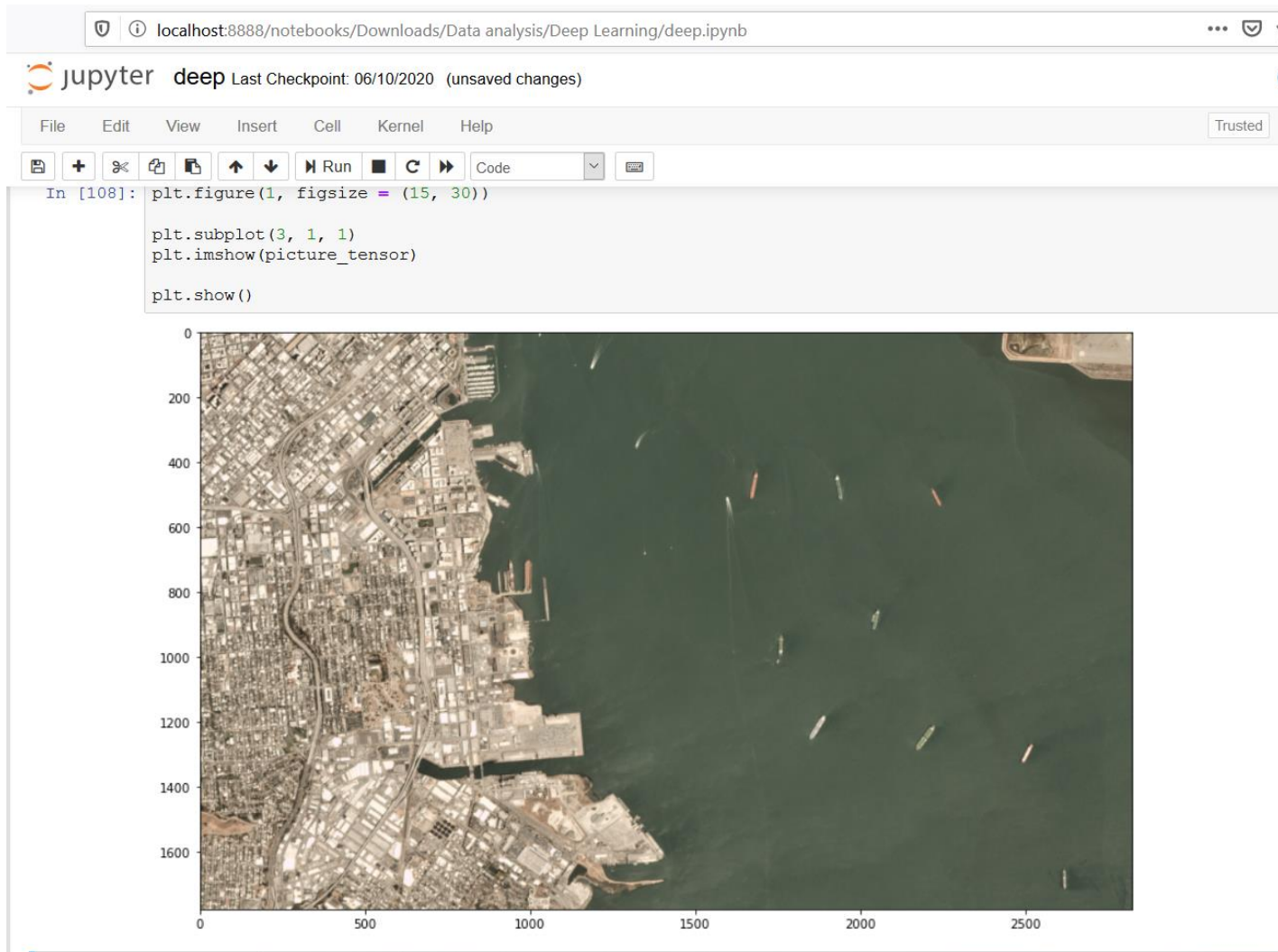


FIG 5.2 INPUT GIVEN TO ML MODEL

While scanning the input image for ship detection, the ML model takes into consideration various parts of input image which might be a ship. The following figures show various parts of the image that the model has predicted to be a ship:

X:1640 Y:410 [[0.01302521 0.98697484]]

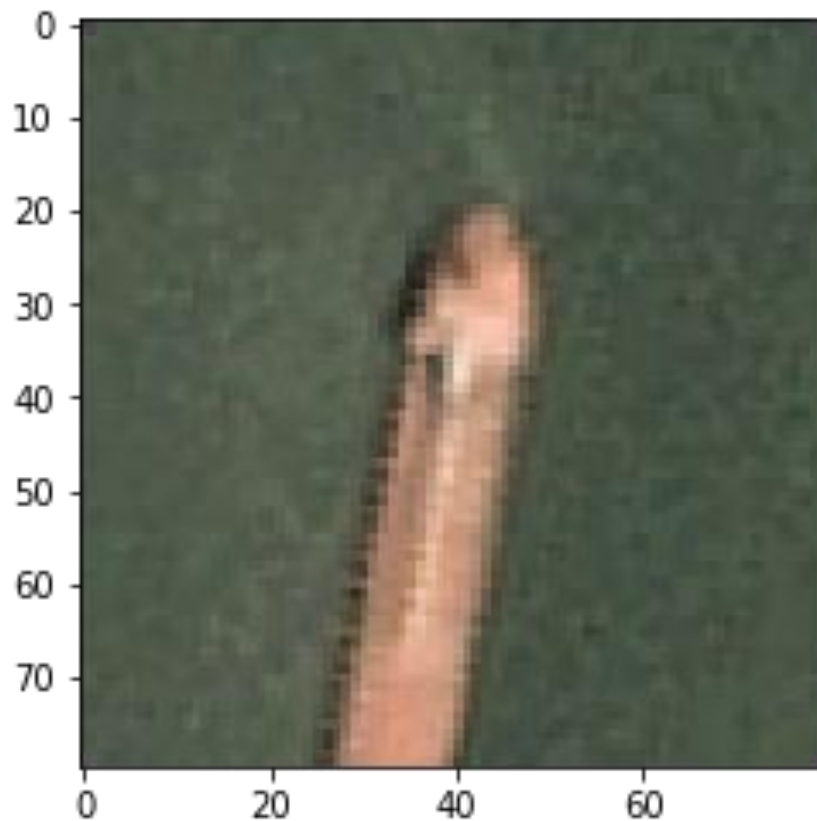


FIG: 5.3 FIRST PREDICTED SHIP

The first prediction is correct as the predicted object is a ship.

X:1890 Y:420 [[0.00718837 0.99281162]]



FIG: 5.4 SECOND PREDICTED SHIP

The second prediction is correct as the predicted object is a ship.

X:2190 Y:460 [[6.14996883e-04 9.99384999e-01]]

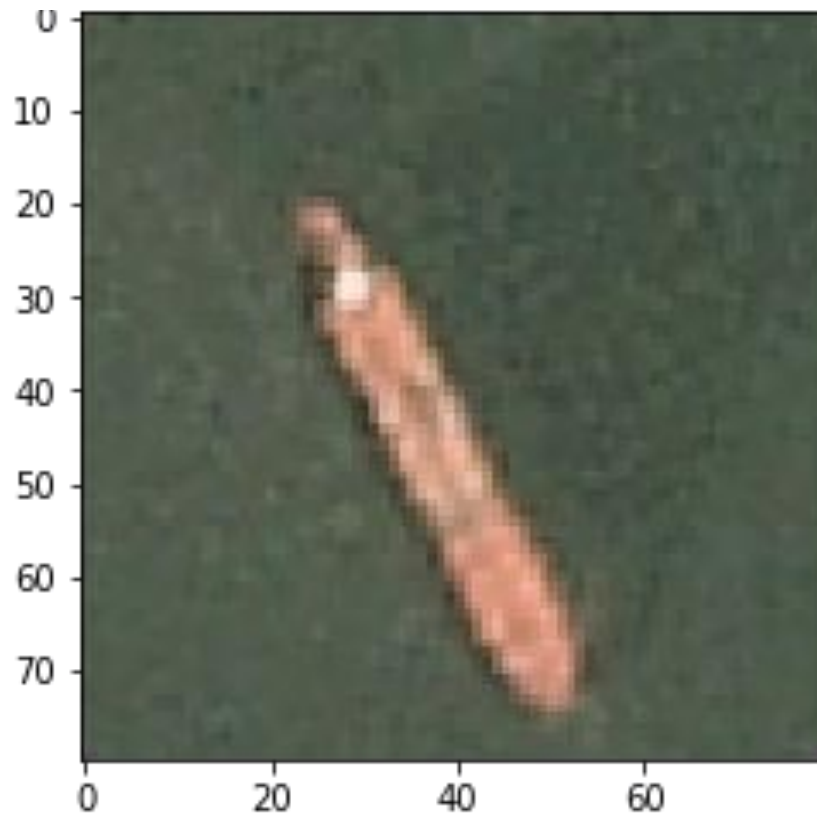


FIG: 5.5 THIRD PREDICTED SHIP

The third prediction is correct as the predicted object is a ship.

X:860 Y:480 [[0.03588928 0.96411073]]

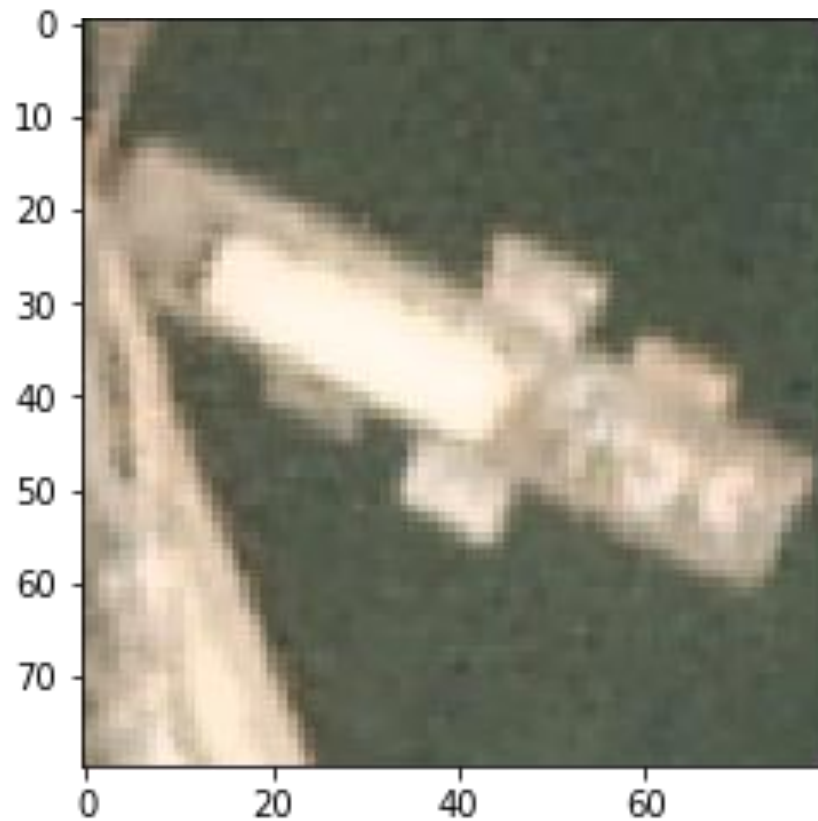


FIG: 5.6 FOURTH PREDICTED SHIP

The fourth prediction is incorrect as the predicted object is not a ship.

X:950 Y:690 [[0.00453715 0.99546283]]

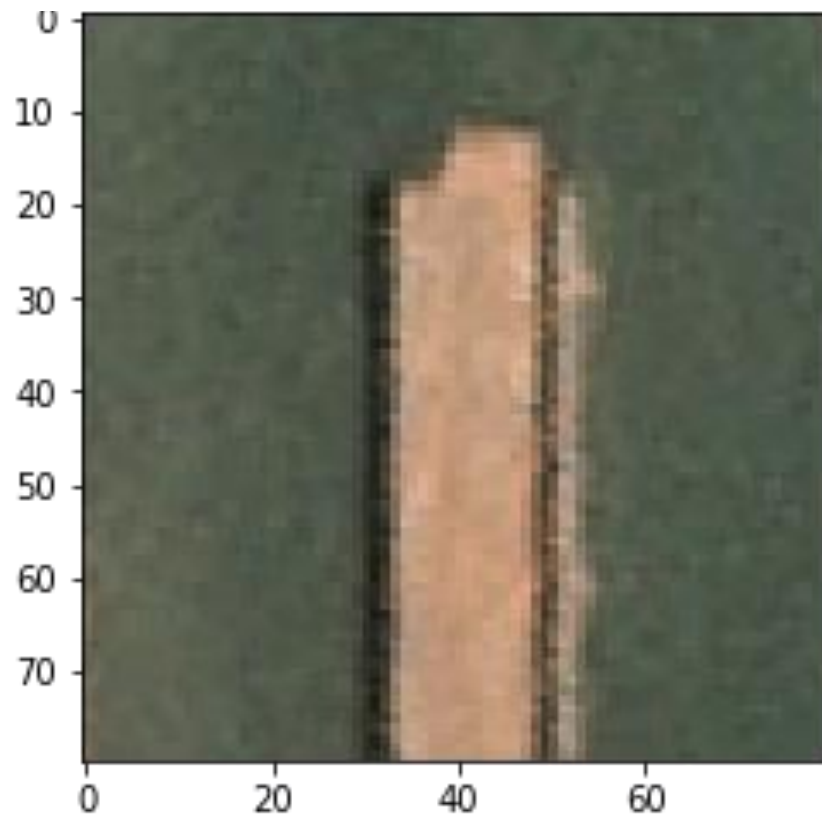


FIG: 5.7 FIFTH PREDICTED SHIP

The fifth prediction is incorrect as the predicted object is not a ship.

X:2000 Y:840 [[0.02599814 0.97400188]]



FIG: 5.8 SIXTH PREDICTED SHIP

The sixth prediction is correct as the predicted object is a ship.

X:1720 Y:920 [[0.0157946 0.98420542]]

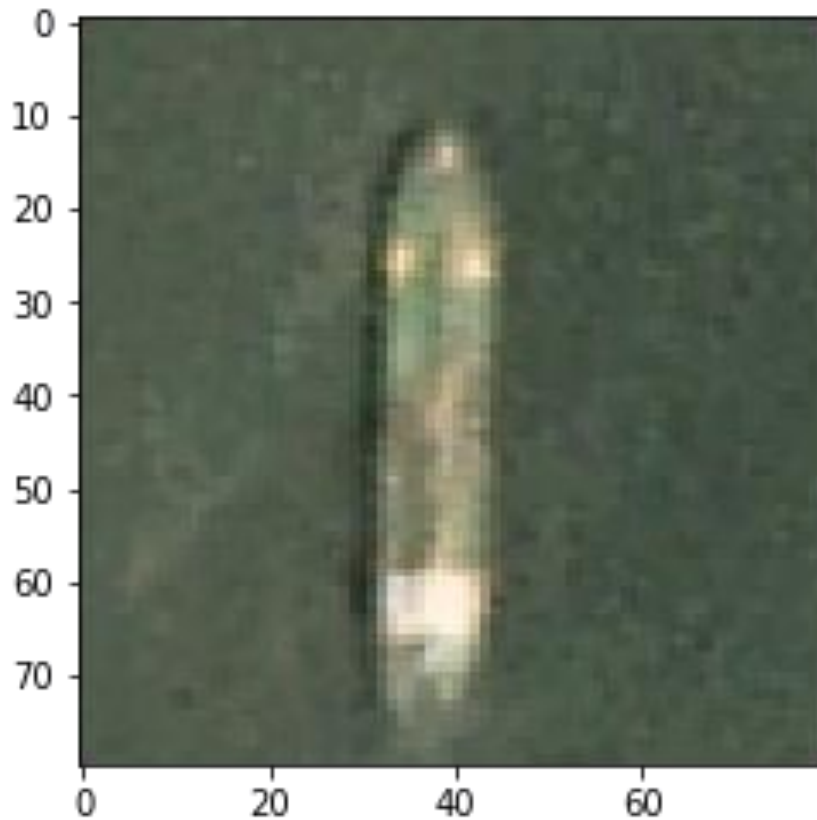


FIG: 5.9 SEVENTH PREDICTED SHIP

The seventh prediction is correct as the predicted object is a ship.

X:1830 Y:1160 [[0.05157557 0.9484244]]

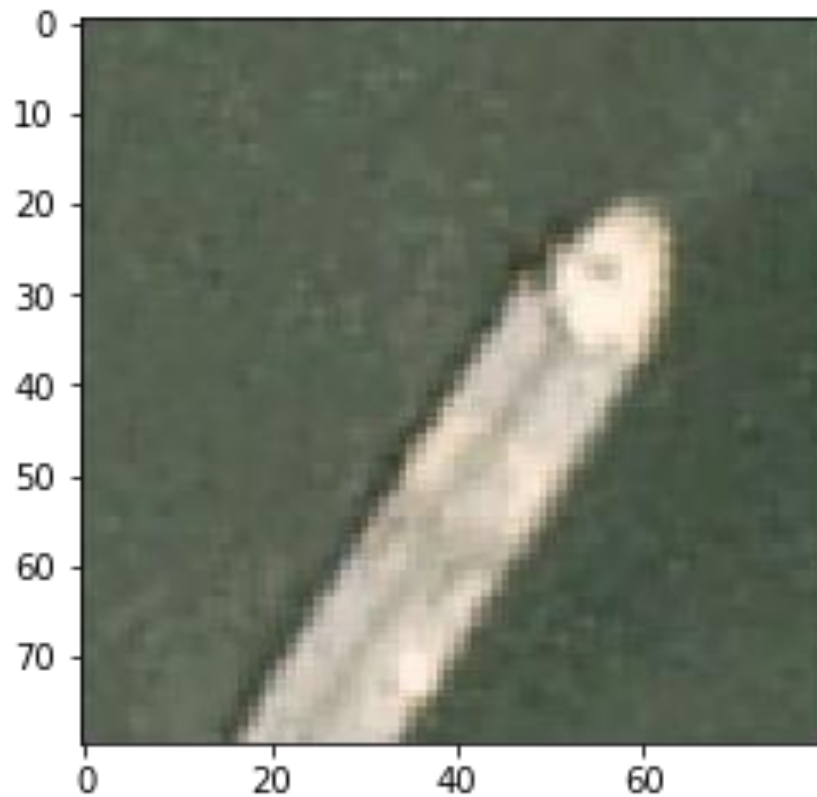


FIG: 5.10 EIGHTH PREDICTED SHIP

The eighth prediction is correct as the predicted object is a ship.

X:2160 Y:1200 [[0.01420313 0.98579687]]

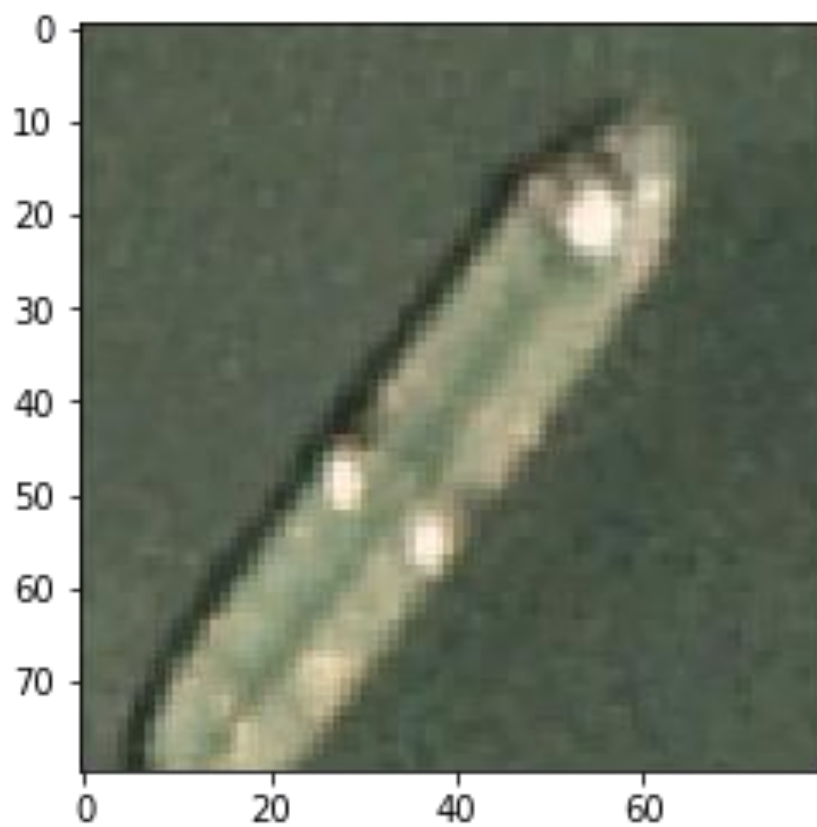


FIG: 5.11 NINTH PREDICTED SHIP

The ninth prediction is correct as the predicted object is a ship.

X:2460 Y:1250 [[0.01058735 0.98941272]]

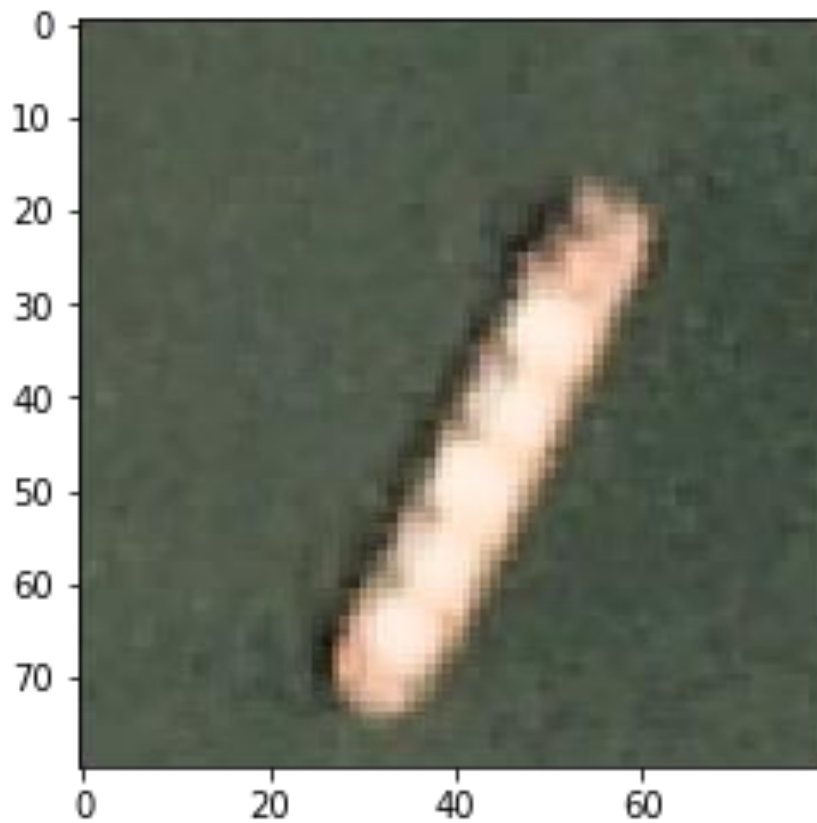


FIG: 5.12 TENTH PREDICTED SHIP

The tenth prediction is correct as the predicted object is a ship.

X:2580 Y:1640 [[0.05226361 0.94773638]]



FIG: 5.13 ELEVENTH PREDICTED SHIP

The eleventh prediction is correct as the predicted object is a ship.

X:1280 Y:1650 [[0.08889534 0.91110468]]

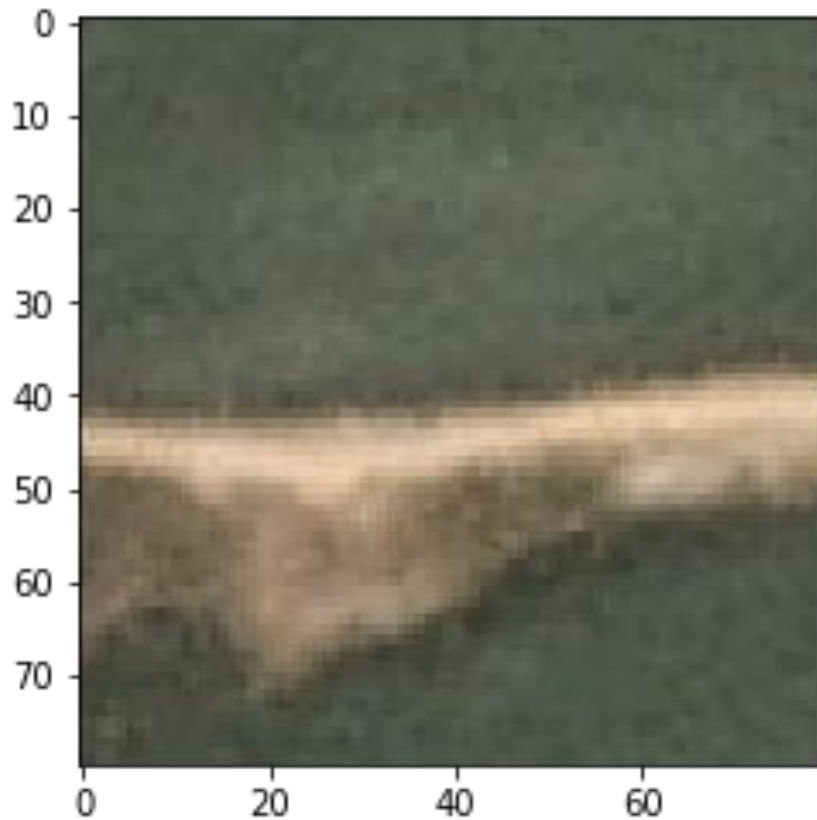


FIG: 5.14 TWELFTH PREDICTED SHIP

The twelfth prediction is incorrect as the predicted object is not a ship.

Figure 5.15 shows the final result with all the ships marked in the satellite image.

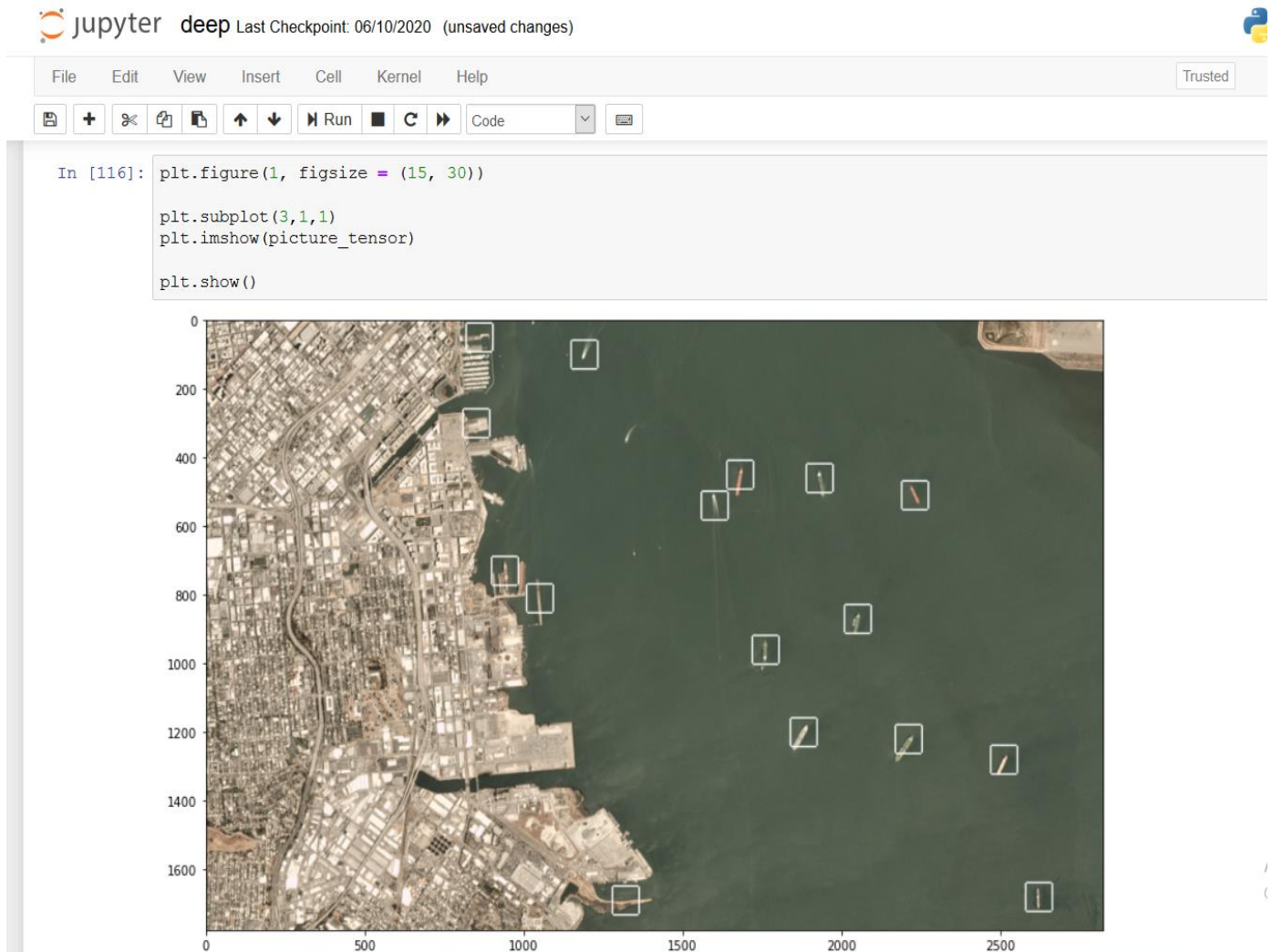


FIG: 5.15 FINAL RESULT WITH ALL MARKED SHIP

The accuracy of the model 85-90%.



CONCLUSION

6.1 Conclusion:

Ship detection using satellite imagery machine learning model requires only a computer system with good processing speed or a good internet connection if you are processing online on platform like Google Colab. The function of entire model is very simple. We train the model with 4000 labelled pictures in which 1000 images are labelled with “ships”-label while 3000 images were labelled with “no ship” label which contains partial ships, land etc. The number of training data can be increased to increase the efficiency and accuracy. After that a satellite image can be provided to the model to detect the number of ships present in that scene. Further, automating this process can be applied to many issues including monitoring port activity levels and supply chain analysis, by applying our ship detection model we will be able to identify any incoming ships using live satellite imagery so that the port activities can be scheduled accordingly. This efficient method can replace previous manual systems and other technologies of performing a difficult task of uniquely identifying a ship or vessel in the vast ocean. This model is more efficient in terms of memory and complexity and in terms of performance; CNN’s outperform Neural Networks on conventional image recognition tasks and many other tasks. If implemented properly this project could be a great use to the future perspective in terms of technology. This method is secure enough, reliable and available for use. No need for specialized hardware for installing the system in the office. It can be constructed using only a computer with fast processing speed.

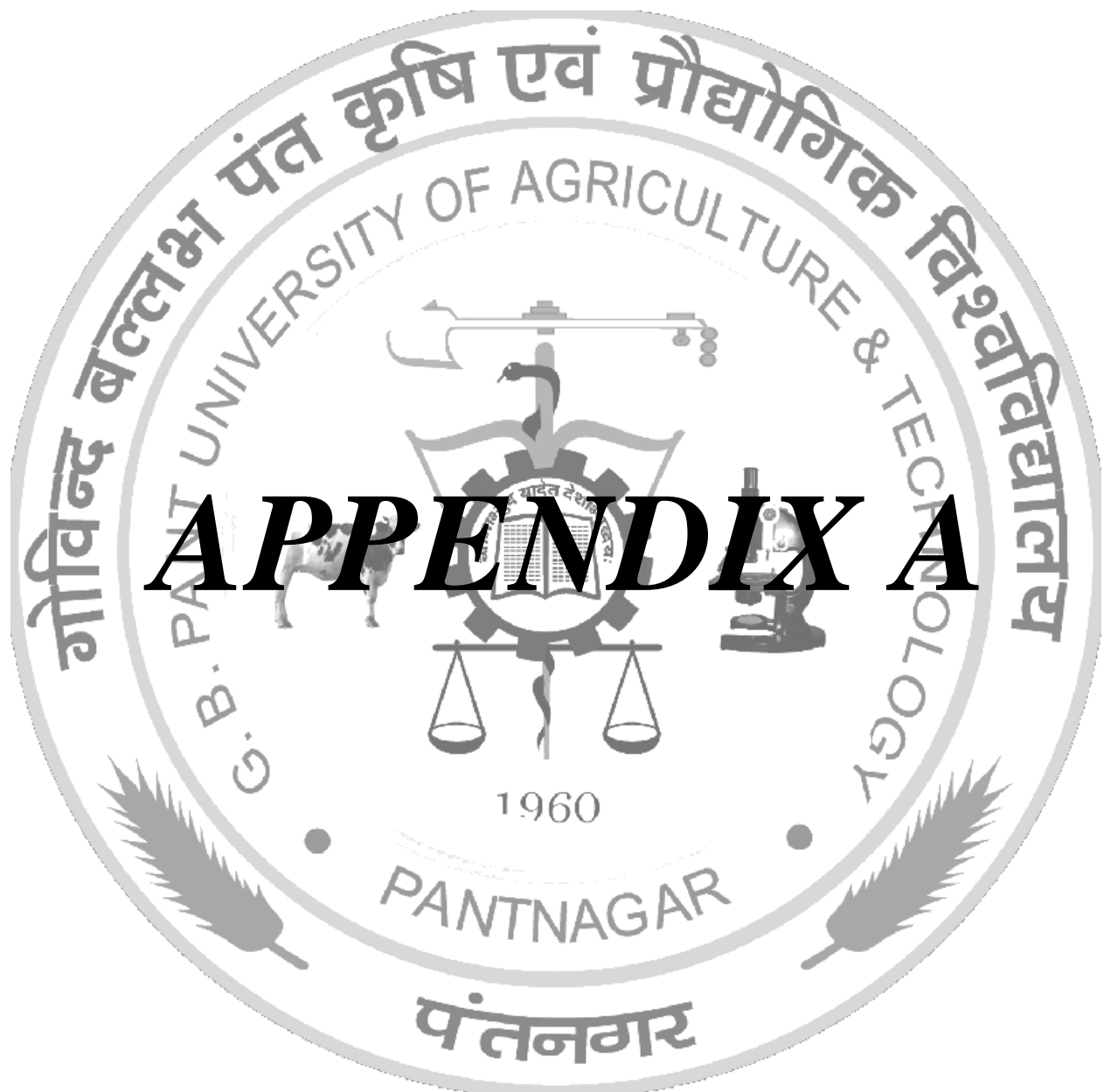
6.2 Future scope:

- To develop an automating process for maritime security and monitoring port level activity.
- Image recognition module can be improved to get faster and more accurate results.



REFERENCES

- Planet's Open_California dataset, which is openly licensed. As such, this dataset is also available under the same CC-BY-SA license.
- R.Binns, "Fairness in machine learning: Lessons from political philosophy," arXiv preprint arXiv:1712.03586, 2017.
- S.Corbett-Davies and S. Goel, "The measure and mismeasure of fairness: A critical review of fair machine learning," arXiv preprint arXiv:1808.00023, 2018
- Kanjir, Urška, Harm Greidanus, and Krištof Oštir. "Vessel detection and classification from spaceborne optical images: A literature survey." *Remote sensing of environment* 207 (2018): 1-26.
- Gu, Jiuxiang, Zhenhua Wang, Jason Kuen, Lianyang Ma, Amir Shahroudy, Bing Shuai, Ting Liu et al. "Recent advances in convolutional neural networks." *Pattern Recognition* 77 (2018): 354-377.
- Sharma, Pranjal, Ujjwal K. Gupta, Markand Oza, and Shashikant Sharma. "Reception-A Deep Learning Based Hybrid Residual Network." In 2019 6th Swiss Conference on Data Science (SDS), pp. 125-129. IEEE, 2019.
- Bayar, Belhassen, and Matthew C. Stamm. "A deep learning approach to universal image manipulation detection using a new convolutional layer." In *Proceedings of the 4th ACM Workshop on Information Hiding and Multimedia Security*, pp. 5-10. 2016.
- Zhou, Lubing. "Correlation filters and feature extraction for facial image analysis." PhD diss., 2014.
- Ullah, Farhan, Junfeng Wang, Sohail Jabbar, Fadi Al-Turjman, and Mamoun Alazab. "Source Code Authorship Attribution Using Hybrid Approach of Program Dependence Graph and Deep Learning Model." *IEEE Access* 7 (2019): 141987-141999.
- Ketkar, Nikhil. "Stochastic gradient descent." In *Deep learning with Python*, pp. 113-132. Apress, Berkeley, CA, 2017.
- https://www.researchgate.net/publication/260622507_Ship_Detection_From_Optical_Satellite_Images_Based_on_Sea_Surface_Analysis
- <https://www.kaggle.com/rhammell/ships-in-satellite-imagery>



PROJECT SOURCE CODE

```
import json, sys, random
import numpy as np

from keras.models import Sequential
from keras.layers import Dense, Flatten, Activation
from keras.layers import Dropout
from keras.layers.convolutional import Conv2D, MaxPooling2D
from keras.utils import np_utils
from keras.optimizers import SGD
import keras.callbacks

from PIL import Image, ImageDraw
from matplotlib import pyplot as plt

f = open(r'../input/ships-in-satellite-imagery/shipsnet.json')
dataset = json.load(f)
f.close()

input_data = np.array(dataset['data']).astype('uint8')
output_data = np.array(dataset['labels']).astype('uint8')
input_data.shape
n_spectrum = 3
weight = 80
height = 80
X = input_data.reshape([-1, n_spectrum, weight, height])
X[0].shape

# get one chanel

pic = X[0]
rad_spectrum = pic[0]
green_spectrum = pic[1]
blue_spectrum = pic[2]
plt.figure(2, figsize = (5*3, 5*1))
plt.set_cmap('jet')
```

show each channel

```
plt.subplot(1, 3, 1)  
plt.imshow(rad_spectrum)
```

```
plt.subplot(1, 3, 2)  
plt.imshow(green_spectrum)
```

```
plt.subplot(1, 3, 3)  
plt.imshow(blue_spectrum)
```

```
plt.show()  
output_data.shape  
output_data
```

```
np.bincount(output_data)
```

output encoding

```
y = np_utils.to_categorical(output_data, 2)
```

shuffle all indexes

```
indexes = np.arange(2800)  
np.random.shuffle(indexes)  
X_train = X[indexes].transpose([0,2,3,1])  
y_train = y[indexes]
```

In [17]:

normalization

```
X_train = X_train / 255  
np.random.seed(42)
```

network design

```
model = Sequential()
model.add(Conv2D(32, (3, 3), padding='same', input_shape=(80, 80, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2))) #40x40
model.add(Dropout(0.25))

model.add(Conv2D(32, (3, 3), padding='same', activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2))) #20x20
model.add(Dropout(0.25))

model.add(Conv2D(32, (3, 3), padding='same', activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2))) #10x10
model.add(Dropout(0.25))

model.add(Conv2D(32, (10, 10), padding='same', activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2))) #5x5
model.add(Dropout(0.25))

model.add(Flatten())
model.add(Dense(512, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(2, activation='softmax'))
```

optimization setup

```
sgd = SGD(lr=0.01, momentum=0.9, nesterov=True)
model.compile(
    loss='categorical_crossentropy',
    optimizer=sgd,
    metrics=['accuracy'])
```

training

```
model.fit(  
    X_train,  
    y_train,  
    batch_size=32,  
    epochs=18,  
    validation_split=0.2,  
    shuffle=True,  
    verbose=2)
```

```
image = Image.open('../input/sfbay/sfbay_1.png')  
pix = image.load()  
n_spectrum = 3  
width = image.size[0]  
height = image.size[1]
```

creat vector

```
picture_vector = []  
for chanel in range(n_spectrum):  
    for y in range(height):  
        for x in range(width):  
            picture_vector.append(pix[x, y][chanel])
```

```
picture_vector = np.array(picture_vector).astype('uint8')  
picture_tensor = picture_vector.reshape([n_spectrum, height, width]).transpose(1, 2, 0)  
plt.figure(1, figsize = (15, 30))
```

```
plt.subplot(3, 1, 1)  
plt.imshow(picture_tensor)
```

```
plt.show()  
picture_tensor = picture_tensor.transpose(2,0,1)
```

```

def cutting(x, y):
    area_study = np.arange(3*80*80).reshape(3, 80, 80)
    for i in range(80):
        for j in range(80):
            area_study[0][i][j] = picture_tensor[0][y+i][x+j]
            area_study[1][i][j] = picture_tensor[1][y+i][x+j]
            area_study[2][i][j] = picture_tensor[2][y+i][x+j]
    area_study = area_study.reshape([-1, 3, 80, 80])
    area_study = area_study.transpose([0,2,3,1])
    area_study = area_study / 255
    sys.stdout.write('\rX:{0} Y:{1} '.format(x, y))
    return area_study


def not_near(x, y, s, coordinates):
    result = True
    for e in coordinates:
        if x+s > e[0][0] and x-s < e[0][0] and y+s > e[0][1] and y-s < e[0][1]:
            result = False
    return result


def show_ship(x, y, acc, thickness=5):
    for i in range(80):
        for ch in range(3):
            for th in range(thickness):
                picture_tensor[ch][y+i][x-th] = -1

    for i in range(80):
        for ch in range(3):
            for th in range(thickness):
                picture_tensor[ch][y+i][x+th+80] = -1

    for i in range(80):
        for ch in range(3):
            for th in range(thickness):
                picture_tensor[ch][y-th][x+i] = -1

    for i in range(80):
        for ch in range(3):
            for th in range(thickness):
                picture_tensor[ch][y+th+80][x+i] = -1

```

```

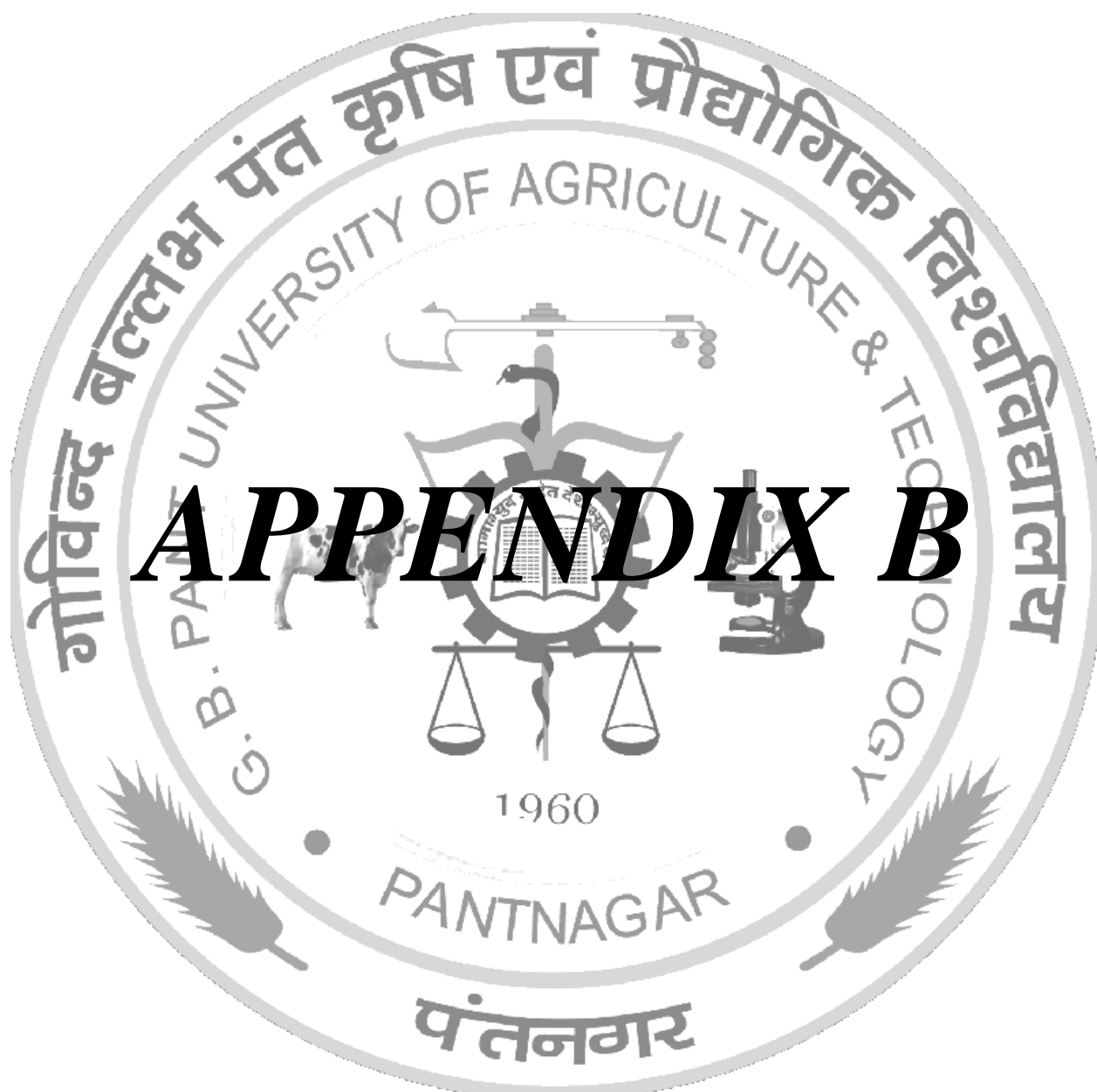
step = 10; coordinates = []
for y in range(int((height-(80-step))/step)):
    for x in range(int((width-(80-step))/step)):
        area = cutting(x*step, y*step)
        result = model.predict(area)
        if result[0][1]> 0.90 and not_near(x*step, y*step, 88, coordinates):
            coordinates.append([x*step, y*step], result)
            print(result)
            plt.imshow(area[0])
            plt.show()

```

```

for e in coordinates:
    show_ship(e[0][0], e[0][1], e[1][0][1])
#picture_tensor = picture_tensor.transpose(2,0,1)
picture_tensor = picture_tensor.transpose(1,2,0)
picture_tensor.shape
plt.figure(1, figsize=(15,30))
plt.subplot(3,1,1)
plt.imshow(picture_tensor)
plt.show()

```



सारांश

विस्तृत फैले समुद्र में जहाज का पता लगाना एक थकाऊ काम है जो इससे पहले मैनुअल रूप से किया जाता था। डिजिटल युग से पहले, कार्गो की आवाजाही और जहाजों को लादने वा उतारने और कस्टम गतिविधियों जैसे पोर्ट गतिविधियों का शेड्यूल मैनुअल रूप से किया जाता था जो एक थकाऊ और समय लेने वाला काम था। इसलिए, इन आने वाले जहाजों का यथाशीघ्र पता लगाने की आवश्यकता थी ताकि पोर्ट प्रबंधक दक्षता बढ़ाने के लिए बंदरगाहों की गतिविधियों को शेड्यूल कर सके और इसीलिए जहाजों और पोर्ट के बीच संबंध स्थापित करने के लिए इस प्रक्रिया में कई तकनीकों का विकास किया गया। फिर प्रौद्योगिकी की प्रगति के साथ जहाजों की निगरानी के लिए विभिन्न तरीके विकसित किए गए। एस.ए.आर (सिंथेटिक एपर्चर रडार) अभी भी समुद्री निगरानी के लिए अग्रणी तकनीक है। पोत पहचान और संचार आमतौर पर स्वचालित पहचान प्रणाली (ए.आई.एस) के उपयोग के माध्यम से किया जाता है, जो जहाज के स्थान, गंतव्य को वायरलेस रूप से प्रसारित करने और अन्य जहाजों और भूमि आधारित प्रणालियों पर पास के रिसीवर उपकरणों की पहचान करने के लिए वीएचएफ रेडियो आवृत्तियों का उपयोग करता है।

कभी-कभी खराब मौसम के कारण, कुछ जहाज खोए हुए कनेक्शन के कारण या किसी अन्य तकनीकी विफलता के कारण विस्तृत प्रसार समुद्र में खो जाते हैं। उस समय इन जहाजों का पता लगाना एक थकाऊ काम बन जाता है और यह बिल्कुल आसान नहीं होता है और हमें उपग्रह चित्रों की मदद से समुद्र के एक बड़े हिस्से को स्कैन करना होता है। नई इमेजरी की बाढ़ संगठनों के लिए मैनुअल रूप से कैप्चर की गई प्रत्येक छवि को देखने की क्षमता को घटा रही है, और विश्लेषण प्रक्रिया को स्वचालित करने में मदद करने के लिए मशीन लर्निंग और कंप्यूटर विज़न एल्गोरिदम की आवश्यकता है, और हमारा शिप डिटेक्शन मॉडल इस स्वचालन को प्रदान करता है।

सैटेलाइट इमेजरी का उपयोग करके जहाज का पता लगाना एक मशीन लर्निंग मॉडल है जिसका उद्देश्य लाइव सैटेलाइट इमेजरी का उपयोग करके समुद्री बंदरगाहों के आसपास जहाजों का पता लगाना है। इस मॉडल का उद्देश्य उपग्रह चित्रों में बड़े जहाजों के स्थान का पता लगाने के कठिन कार्य को संबोधित करने में मदद करना है। इसके अलावा, इस प्रक्रिया को स्वचालित करने से पोर्ट गतिविधि स्तर की निगरानी और आपूर्ति श्रृंखला विश्लेषण सहित कई मुद्दों पर लागू किया जा सकता है, हमारे जहाज का पता लगाने वाले मॉडल को लागू करके हम लाइव उपग्रह इमेजरी का उपयोग करके किसी भी आने वाले जहाजों की पहचान करने में सक्षम होंगे ताकि बंदरगाह गतिविधियों को तदनुसार निर्धारित किया जा सके।



VITAE

VITAE



The author, **Shashank Diwakar** was born on **5th October, 1997** at Kanpur, Uttar Pradesh. He passed his High school and Intermediate from **Campus School, Pantnagar** during the year 2013 and 2015 respectively. He joined as an under graduate student in **College of Technology, Govind Ballabh Pant University of Agriculture and Technology, Pantnagar** for programme **Bachelor of Technology(Information Technology)**.

Address for Communication

Shashank Diwakar
S/O Mr. D.K.Diwakar
Gandhi Enclave, Pantnagar
U.S.Nagar, Uttarakhand
Pin code-263145
Phone no.- +91-7055670919
Email- shashankdiwakar00@gmail.com

VITAE



The author, **Akshay Nagpal** was born on **22nd November, 1998** at Kichha, Uttarakhand. He passed his High school and Intermediate from **Jaycees Public School, Rudrapur** during the year 2014 and 2016 respectively. He joined as an under graduate student in **College of Technology, Govind Ballabh Pant University of Agriculture and Technology, Pantnagar** for programme **Bachelor of Technology(Information Technology)**.

Address for Communication

Akshay Nagpal
S/O Mr. Vimal Kumar Nagpal
H.No 97, Near P.W.D Guest House
Punjabi Colony, Kichha, U.S.Nagar, Uttarakhand
Pin code-263148
Phone no.- +91-8476080084
Email- akshay.ngpl10@gmail.com

VITAE



The authoress, **Ankita Satyarthi** was born on **28th July, 1997** at Haldwani, Uttarakhand. She passed her High school from **Jim Corbett International School** and Intermediate from **Nirmala Convent Senior Secondary School, Haldwani** during the year 2013 and 2015 respectively.

She joined as an under graduate student in **College of Technology, Govind Ballabh Pant University of Agriculture and Technology, Pantnagar** for programme **Bachelor of Technology(Information Technology)**.

Address for Communication

Ankita Satyarthi
D/O Mr. Nandlal Satyarthi
Berikhatta Hydel Gate
Kathgodam, Nanital, Uttarakhand
Pin code-263126
Phone no.- +91-9536247269
Email- ankitasatyarthi17@gmail.com

VITAE



The author, **Sarthak Mittal** was born on **23rd November, 1997** at Dehradun, Uttarakhand. He passed his High school and Intermediate from **St. Thomas College, Dehradun** during the year 2013 and 2015 respectively. He joined as an under graduate student in **College of Technology, Govind Ballabh Pant University of Agriculture and Technology, Pantnagar** for programme **Bachelor of Technology(Information Technology)**.

Address for Communication

Sarthak Mittal
S/O Mr. Neeraj Mittal
13/19 P.D Tandon Marg
Laxman Chowk, Dehradun, Uttarakhand
Pin code-248001
Phone no.- +91-7906718650
Email- sarthakmittal961@gmail.com