Technische Universität Berlin
Faculty:
Geodesy and Geoinformation Science

Professor:
Dr. Martin Kada

Master Thesis

Master Thesis:

Lunar Crater Detection via Deep Learning and its Application for Age Estimation

Author:
Tahir, Waqas

Spervisors:
Agoub, Amgad
Gläser, Philipp

January 15, 2020

Contents

1 Introduction

Studying impact craters present a valuable information about the geology of planets and characteristics of the surface. Computation of crater density has provided a gateway for the establishment of evolution of planet terrains chronologically [Martins et al., 2009]. This study provide not only the information about geological processes of the planet but also the history of our solar system. According to geologists impact craters are the only modifying and surface forming processes for other planets and moons of all planets [Koeberl, 1994]. Impact craters have also been a useful source of information for planetary scientists in providing the relative age of surfaces. If the relationship between crater size and impact energy is known and flux caused by the impact is understood then craters with larger densities indicate older surfaces [Ivanov, 2002].

Various processes alter crater populations especially of those with smaller diameters [Opik, 1965] more often in planets with dense atmosphere resulting in deviations in the crater size-frequency distributions (CSFD). Such processes are: deceleration, ablation, fragmentation of meteors while passing through the atmosphere prior to surface impact and postformation modification of craters by erosion and deposition. Therefore, crater counts with relatively smaller diameter (i.e. diameter < 0.01 km) are at a larger risk of representing an age which could be misinterpreted if the modeled production function does not take into account the factors responsible for altering CSFD in an observed range of diameter [Hartmann, 1981]. Moreover, several factors may lead to surface age as well as statistical uncertainties because smaller crater's identification is prone to certain biases such as resolution limits, illumination effects, compact crater count areas or limited number of craters [Soderblom, 1970].

In case of earth's Moon, modeling of impact craters depends on the knowledge of age-dated lunar samples with correlation to observed CSFDs [Williams et al., 2018]. Chronology is a system composed of two elements such as a production function describing the CSFD shape and a chronology function which is related to the accumulation of craters density to absolute time. Both of these functions collectively provide a predicted CSFD or isochron for a given time length a surface has been under crater strikes. This is valid for lunar surface as Moon has no atmosphere. This chronology can also be applied to other solar system objects but with an addition of factors such as impactor flux and surface gravity [Ivanov, 2002].

If CSFD and the diameter of crater is known then age of the lunar surface can be determined using both of these parameters. It is only possible to determine the age of the same area where crater count has been performed. There are production functions proposed by planetary scientists which provide an approximation to surface age if CSFD and crater diameters are known. The examples of such functions are Hartmann Production Function (HPF) and Neukum Production Function (NPF).

Crater counts provides the frequency of crater distribution of a certain area. This can be performed by counting craters manually. This task could not only be time consuming but also expensive and labor intensive depending upon the number of images and amount of craters present in images. Satellites are the source of images which provide opportunity to count craters depending on the type of cameras on board. Also number of craters in a certain area may change with a passage of time which would mean that possibly same task for the same surface has to be performed again.

Since early 2000 many missions and satellites have been deployed to Moon for research purposes including crater studies. LRO (Lunar Reconnaissance Orbiter) is one of the satellites on lunar mission operated by USA and is collecting lunar images since 2009. LRO has gathered millions of images since its launch. Lunar Reconnaissance Orbiter Camera (LROC) was specifically designed for the assessment of meter-scale and smaller features to help carry out safety analysis for future lunar landing sites on moon including polar region. The Narrow Angle Cameras (NAC) mounted on LRO are able to detect craters with diameter of 2.5m or greater [Robinson et al., 2010]. It is common to find craters less than 100m in diameter on lunar surface [Robinson et al., 2010]. Now with this information these images can be utilized to count craters. Traditional methods of counting craters are manual by visual inspection of images. This approach is not practical when dealing large amount of images with craters of various sizes on either Moon or any other planet. But to achieve this task through computers there is a demanding processing hardware requirement.

Images taken from LRO are large in size (resolution of 1.009). The data is available publicly and can be seen or downloaded through this link *(http://wms.lroc.asu.edu/lroc/)*. Such images cannot be processed without a competitive machine. There are methods developed to perform image processing tasks which are highly dependent on a machine's memory and performance. Therefore, developing techniques without advancement in computer technology was simply not enough. This problem was already known during 1960s and therefore it lead to the development of Graphical processing unit (GPU). A GPU performs quick math calculations and frees the space for CPU to do other tasks. Unlike CPU it has thousands of cores designed for multi-tasking. A much needed hardware to perform computationally expensive calculations without exhaustion.

In 2000s many companies like Intel, Nvidia and AMD/ATI stepped into the race of manufacturing faster GPUs and dominated market. This competition continues till today and GPUs are becoming more and more powerful. Meanwhile, the need for GPUs is also increasing because of big data processing needs. Nvidia introduced a chip capable of programmable pixel shaders i.e. compute color, position, depth or stencil of a pixel, the Ge-Force 3. A short program could now process each pixel before projecting it to the screen, this processing included but not limited to addition of image textures. By 2002 ATI Radeon 9700 the world's first Direct3D 9.0 accelerator was introduced by Microsoft containing support for multiple render targets, floating point texture formats, multiple-element textures and stencil buffer methods. In 2010 Nvidia began a partnership with Audi to power car dashboards. This mainly increased the functionality of navigation and entertainment systems. In 2014 the PS4 and Xbox One were released powered by GPUs based on AMD's Radeon HD 7850 and 7790. Lately RTX was released by Nvidia with the aim of enabling real time ray tracing. This was a new development in computer graphics for generation of interactive images reacting to lighting, reflections and shadows. RTX also includes artificial intelligence (AI) integration like enhancing video processing, graphics and images directly into applications. Today, parallel GPUs are making complex computations in the fields of oil exploration, machine learning, image processing, 3D reconstruction, statistics and even in stock market for stock rates determination.

With this advancement in computer processing ability, complex computation tasks are now possible to perform. That means a human labor intensive task of counting objects can possibly be automated but the problem of detecting objects could be complicated specially when object background is complex. With advancements in computer technology it was no longer optimum to rely on manual detection of objects. With big data and challenge of solving tasks in limited time, it was required to develop a mechanism that could solve such tasks faster as compared to expensive manual operations and meanwhile limit man power. This gave rise to research in deep learning approaches to solve this problem.

In the past automatic detection of craters turned out to be a difficult task in cases where rims were overlapped, not clear or if image was noisy [Sawabe et al., 2006]. Multiple automated methods were presented by [Sawabe et al., 2006] using the data acquired by Clementine and Apollo. One of the methods was to thin down a set of edge pixels to one pixel using Hilditch's thinning algorithm. The lines were then connected depending upon the direction and length. If the resultant lines were closed with roundness more than 0.8 then lines were regarded as crater.

However in last decade, new techniques have been developed in object detection, classification, localization and semantic segmentation. These methods can now put into application because of GPUs of today, which are not only capable of performing computationally intensive tasks but also shortens the amount of processing time as compared to previous versions of GPUs.

1.1 Problem Statement

Based on satellite images characteristics of a planet terrain could be complex that can make craters difficult to identify. Some of the satellite images such as Clementine have noise and that pose challenge for detection of degraded features on a planet surface. With high resolution satellite images (like LRO) opened new horizons of research and in recent decade many approaches have been developed to detect craters. These methods range from generalized Hough transformation, crater development algorithms to deep learning which was mainly applied from other areas (such as medical imaging and computer vision) for craters detection. But the main focus remained on medium to large sized craters on digital elevation models. However, an actual need is to detect smaller crater which are less than a kilometer size, not only located at the smooth surfaces and also present on the part of image with a variable level of illumination. An algorithm should be build to detect smaller craters of few meters located on a varying terrain rather than algorithm only able to detect very large craters and those located on a simple terrain. Small crater detection has also more significance compared to large kilometer sized craters because smaller ones are much shear in number and far less abundant than larger ones and therefore large craters can be detected by visual inspection without a substantial overwhelming human effort. However small craters because of vast quantity requires an automated detection approach. Optical images have complexity of shades and background terrain features, therefore detection of craters directly on optical images was largely neglected. Although high resolution images provide an ease for visual identification of small or partially faded craters but increase in data and overcome the visual inspection effort a need for automatic detection algorithm exist and detection of craters with complex features of terrain, shapes and sizes is a challenging task.

1.2 Research Approach

The approach is to automate crater detection process by using deep learning technique which would be faster as compared to manual counts and does not require involvement of experts for annotation. The model to perform pixelwise segmentation is based on Convolutional Neural Network (CNN). In this model images are subjected to convolutional operation (mathematical element wise-multiplication) and such type of neural networks are known as ConvNets or simply CNNs. These networks belong to category of supervised machine learning that have proved their success by winning Large Scale Visual Recognition Challenge (ILSVRC-2010) competition which was composed of 1.2 million high-resolution images. [Krizhevsky et al., 2012] achieved top-1 and top-5 error rates of 37.5% and 17.0% respectively.

There are many reasons for usage of CNN to crater detection problem. CNNs have proven record in computer vision problems and such datasets where a correlation lies between features [Long et al., 2015]. Not only in computer vision based on images but also CNNs have demonstrated their versatility in sounds and signals as well. Another major reason of using CNNs is the ability to learn from features hence they have the capability to engineer their own representation of features, thus removing human involvement in developing sophisticated pre-processing models and custom input features. CNNs are also able to classify objects in images which are in different scales even when on a single image. For lunar craters this case is similar where craters range from few meters in diameter to several hundreds of meters and could have multiple instances in a single image.

An algorithm based on CNN is trained and tested for lunar crater detection. The inputs are images taken by Lunar Reconnaissance Orbiter irrespective of specific camera limitations. The outcome of this model is a trained version of this model also called as a checkpoint with minimum loss function. This trained model is tested on lunar images which is divided into two groups of testsets, first testset is taken from dataset for training and evaluation split and second is taken from [Polegubic, 2020] with different sun illumination angle and it yields predictions of craters in those images (pixel wise segmentation of crater in an image). These predictions after post-processing provides the size frequency distribution of craters. In post-processing step, several binarization methods are applied to detect blobs from the segmentation heat map. These methods are analyzed and compared to each with respect to F1-score. This measure is also used to evaluate performance of algorithm. In order to detect craters from blobbed images and their respective diameters Region proposal algorithm was applied. Now with this information a lunar production function is used to plot the age of surface where craters are counted.

2 Background

Studying the surface of other planets is a source of learning about our own planet. How crater impacts can effect the life on Earth, climatic change globally and possible consequences of extreme environmental disaster. It is a known fact that solid planet surfaces are covered with tremendous amount of craters of various sizes and shapes [Melosh, 1988]. Largely they form as a result of meteoroid impacts and also considered as windows into the interiors of solid planets [Honda and Azuma, 2000]. Knowledge of lunar surface features including craters is vital for safe landing on Moon [Ivanov et al., 2015]. Finding age from crater-counts is a commonly accepted method. Therefore, crater-counts provide a much less expensive way to find out the age of surface as compared to Radioactive Age-Dating. As for the second, a rock sample is needed and age can only be determined for a specific area whereas by crater-counts it is possible to find out the age of a larger area. Despite the importance of crater and tremendous amount of data available, for decades crater analysis remained dependent on human vision and manual operation [Honda and Azuma, 2000].

Pattern recognition algorithms originally developed for particle physics, medical imaging or optical character recognition were applied on planetary science problems such as crater detection. Types of identified features were ellipses, circles, ridges and lines etc. Hough transformation was developed originally for high energy physics by Hough in 1959. Later, generalized Hough transformation (GHT) methods were developed and applied for crater detection. One of the application of GHT based method was made by [Honda and Azuma, 2000] for selected lunar images taken by Clementine but this method was not validated for scientific use. An ellipse/edge detection method based on GHT was developed by [Leroy et al., 2001]. The efficiency of this method was measured as ratio of detected craters to true craters, which was about 20% and was acceptable for addressing landing problems.

Impact craters remained some of the most studied features in lunar ans planetary science. This made crater detection algorithms or CDAs an important subject of scientific research, as evident form the amount of publications regarding crater detection in the past. This research range from dating of planetary surfaces, searching of still unknown crater impacts on surface of the Earth and safe landing sites on other planets and asteroids. The application of CDAs introduced by [Salamuniccar and Loncaric, 2010] were based on edge detection and elevation data. The CDA presented was an improvement to previous CDAs and the purpose was to contribute to Martian crater catalog. With each new planetary and lunar mission, the volume of data increases significantly. This justifies research on automatic detection methods and importance of their role in processing, archiving, retrieving and interpreting large amounts of image data. [Salamuniccar and Loncaric, 2010] provided an overview of 73 CDA-related publications from numerous authors. Working on CDAs has been a challenging task for many reasons, e.g. multiple possible applications of CDA could provide a good solution for one specific problem on the particular planetary surface i.e. dating [Sawabe et al., 2006] but not necessarily good for other surfaces depending on the terrain features or problems such as autonomous landing on other planets or asteroids [Leroy et al., 2001]. Another challenge of CDA is to distinguish between crater and non-crater objects. Set of features that separates these two depends on the type of surface, properties of illumination and shapes and sizes of craters. Existing algorithms mainly focus on large craters located on simple surfaces which dictates a specific choice of image features [Bandeira et al., 2010]. Therefore, it is not surprising that still no CDA available is robust as scientific community would like and that fits several research groups. Also CDA not only detects craters but also finds crater candidates that needs to be manually rejected, or corrected with diameter by computer-assistance.

Despite of extensive research in CDAs, no algorithm became a standard tool for planetary science practitioners and crater counts continued to be done via visual inspection even as high resolution datasets keep on increasing [Bandeira et al., 2010]. Crater appearance in an optical image depends on their level of degradation, inter morphologies (such as presence of central peaks, central pits, peak rings and wall terraces etc.), amount of overlap on other craters, quality of image (which includes illumination angle and surface properties) and on size that might differ by the order of magnitude. A notable advancement in CDA was proposed by [Bandeira et al., 2010] which used the approach of applying series of filters for background noise removal and creates a set of features that look for the characteristics of crescent-shaped shadow of a crater. In addition to noise removal, texture recognition was added to improve algorithm precision.

Another approach begins with template matching in which image array pixels are rotated, translated or transformed to match pieces of an image. This establishes an image-based paradigm which can be extended to matched spatial features, principle components methods and artificial neural networks [Brunelli and Poggio, 1993]. An image matching algorithm known as The Continuously Scalable Template Matching (CSTM) was implemented and tested by [Burl et al., 2001] on selected lunar images. The algorithm uses templates provided by scientists for generation of a model to detect target in a user specified continuous range of scales. Statistical efficiency of implementation of algorithm was measured on regions of Lunar Maria (images provided by Clementine), which was about 80% with 12% false detection rate. Craters less than 5 pixels were neglected. However, the reduction in performance was caused by complexity in background terrain for crater detection in Europa images.

For crater automation, large focus remained on looking for circular or elliptical shape of edges on crater boundary i.e. Hough transform [Honda and Azuma, 2000]. Boundary based approaches seemed to perform well (relative to image resolution) under certain conditions such as detection of medium to large craters with limited texture in background because of other processes or features. However the performance of these deteriorated with complexity in background terrain or when craters are small. Alternating to boundary-based detection methods [Wetzler et al., 2005] proposed to look directly at the pixel-level pattern in an image. The idea was to automate crater detection process by looking for adjacent bright-dark regions of proper relative size.

Existing approaches to detection techniques can be divided into two categories: supervised (requiring a labeled input data) and unsupervised (fully autonomous). [Stepinski et al., 2009] has discussed both of these approaches and their usage. Unsupervised techniques are based on pattern recognition approaches to identify crater rims in an image as circular objects. Supervised techniques are the machine learning methods to train a classifier which is then used to distinguish between craters and other objects. However, in both approaches features are detected by narrowing down to the set of potential candidates. In a supervised method the narrowing is achieved by thresholding a probability of positive detection ba a classifier. In an unsupervised approach narrowing is achieved by thresholding a parameter that measure how well the object fits a circle.

2.1 Bounding Box Proposal

In object detection, a bounding box (also referred to as region of interest or box proposal) shows the existence of object. It is a rectangular region of the input image containing the object. Bounding boxes can be generated by some heuristic search methods such as finding region proposal by objectness, region proposal network (RPN) or by selective search method. A bounding box can be either represented by storing its two corner coordinates (x_0, y_0, x_1, y_1) or most commonly by storing its center location along with width and height such (x, y, w, h). A bounding box is generated on the basis of confidence score that how likely an object exists inside the box. The difference between two bounding boxes is usually measured as the L2 distance of their vector representations. Another simplest way is to compare the images by taking pixelwise difference and summing up all the differences. To perform such procedure, given two images can be represented as vectors I_1, I_2 then L1 can be computed as:

$$d_1(I_1, I_2) = \sum_p |I_1^p - I_2^p|$$

$$\left| \underset{\text{test image}}{\begin{bmatrix} 56 & 32 & 10 & 18 \\ 90 & 23 & 128 & 133 \\ 24 & 26 & 178 & 200 \\ 2 & 0 & 255 & 220 \end{bmatrix}} - \underset{\text{training image}}{\begin{bmatrix} 10 & 20 & 24 & 17 \\ 8 & 10 & 89 & 100 \\ 12 & 16 & 178 & 170 \\ 4 & 32 & 233 & 112 \end{bmatrix}} \right| = \underset{\text{pixel-wise absolute value differences}}{\begin{bmatrix} 46 & 12 & 14 & 1 \\ 82 & 13 & 39 & 33 \\ 12 & 10 & 0 & 30 \\ 2 & 32 & 22 & 108 \end{bmatrix}} \rightarrow 456$$

Figure 1: Example of two images which are represented as vectors and pixelwise difference is calculated to compare both images with L1 distance. This example shows one color channel. All the pixel-wise differences are added to denote a single digit value. If this value is close to zero then it indicates that images are identical. A large value shows that both images are very different.

The difference between bounding boxes can also be measured by the L2 distance. Similar like L1, images are represented in a vector form. w and h can be log-transformed before calculating distance. L2 has the geometric representation of computing euclidean distance between two vectors as:

$$d_2(I_1, I_2) = \sqrt{\sum_p (I_1^p - I_2^p)^2}$$

In simple words this operation also computes the pixelwise difference like L1 but here all the differences are squared then added and finally equation is subjected to square root. Difference in measurement between both metrics differs in a way that L2 prefers many medium disagreements to one big one when it comes to difference between vectors.

In case of multiple bounding boxes a common algorithm to merge them is non maximum suppression (NMS). A bounding box that overlaps another one having higher confidence score with a factor such as its intersection over union (IoU) is greater than IoU threshold, is removed. Intersection over union provides similarity between two bounding boxes. $Area\ of\ Overlap/Area\ of\ Union = IoU$. The bounding boxes play an important role in setting up dataset for the algorithm. Learning of the algorithm largely depends on correctly placed bounding box on the object.

With sliding windows, a bounding box can be attained but it may not properly fit the object. With bounding box and stride size, it might be only able to cover a part of the object and not the complete object. This problem can be solved using an approach developed by [Redmon et al.,] where a picture is divided into multiple grids and an image classification and localization algorithm is applied to each grid. Every grid has a label y which represents some parameters as shown below:

$$y = \begin{bmatrix} p_c \\ b_x \\ b_y \\ b_h \\ b_w \\ c_1 \end{bmatrix}$$

where p_c is whether or not there is an image, b_x, b_y, b_h and b_w is to specify bounding box, c_1 is object class (i.e, craters). If there is no crater then p_c will be zero and hence the entire vector is to be dropped. When $p_c = 1$, it represents there is a crater and all the bounding box parameters would specify position of this box and hence the value of c_1 would be 1. Each grid cell will have this output vector y. The idea is to feed an input image and run forward pass (convolution, max pooling and relu) to get this output vector y.

The evaluation of object detection algorithm can be performed using IoU. As mentioned before, it gives the ratio between area of overlap and area of union. So if the algorithm outputs a bounding box which does not properly fit the ground truth bounding box representing the object then by convention the answer is taken correct if $IoU \geq 0.5$. If predicted and ground truth bounding boxes overlaps perfectly then IoU would be 1. The higher the IoU, the more accurate the bounding box is.

2.2 Normalization

In machine learning normalization is a typical practice in preprocessing of data before its final preparation to run it through a neural network. Data can be standardized meaning subtracting a mean of dataset from each data point and dividing the dataset by standard deviation, mathematically standardization z written as:

$$z = \frac{x - m}{s}$$

Another way of normalization is to normalize a dataset by bringing it in a range of 0 to 1. This is typically true for image processing when pixels ranging from 0 to 255 which are normalized from 0 to 1. It is necessary to normalize data because without it some numerical data points can be very high and other might be very low. The larger data points in non-normalized datasets can cause instability in neural networks because the relatively large inputs can cascade down through the layers in the network which may cause imbalance gradients which may therefore cause the exploding gradient problem. This makes a network drastically harder to train and additionally significantly decrease training speed.

In neural network exploding gradient refers to a problem when weight is larger than identity and it is multiplied by pixels of image (ignoring bias in this case) then resultant is very high, this will further pass on to next neuron and it will be even higher, thus in a network specially in a deep network it will result into a very large value. The opposite problem is a vanishing gradient when a weight is less than identity then multiplication of weight results into even a smaller number therefore output is a very small value which means that network has barely learned in all of the layers.

2.3 Backpropagation

The CNN requires to update its weight for a given training data in order to reduce loss. Backpropagation is an efficient method for computation of gradients which are required for gradient based optimization of weights or kernel parameters in neural networks [Rumelhart et al., 1988]. This optimization problem refers to minimize the loss function which is performed by the specific combination of weights. Backpropagation requires the computation of loss function at each iteration therefore loss function should be differentiable and continuous.

For a multiplication function of two numbers i.e. $f(x, y) = xy$, it is possible to derive the partial derivative for either of input:

$$f(x,y) = xy \quad \rightarrow \quad \frac{\partial f}{\partial x} = y \quad \frac{\partial f}{\partial y} = x \tag{1}$$

The derivation function of a variable indicate the rate of change of a function with respect to that variable surrounding an infinitely small region near a specific point.

$$\frac{df(x)}{dx} = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h} \tag{2}$$

In above equation the operator $\frac{d}{dx}$ (a derivative operator) is applied to a function f. The resultant is a derivative. It can be interpreted as; if h is very small then s straight line determines approximation of function and slope of the line is derivative. The sensitivity of an expression on a value is determined by the derivative. If $x = 4, y = -3$ then $f(x, y) = -12$ and this would return a derivative on $x\frac{\partial f}{\partial x} = -3$. It is clear that if value of this variable is increased by a tiny amount, it would create an impact of three times decrement because of its negative sign. It also be seen by rearranging the above equation such that; $f(x + h) = f(x) + h\frac{df(x)}{dx}\Big)$. In other way as $\frac{\partial f}{\partial y} = 4$, it is also expected that by increasing the value of y by a tiny amount which is h would also increase the function output by $4h$ because of positive sign.

The derivative of function $f(x, y)$ is a vector of partial derivatives which is $\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right] = [y, x]$. Gradients can also be calculated for addition operation such as:

$$f(x,y) = x + y \quad \rightarrow \quad \frac{\partial f}{\partial x} = 1 \quad \frac{\partial f}{\partial y} = 1 \tag{3}$$

This indicates that derivative of both x, y is one regardless of values of x, y. Increasing either x or y will increase the value of output to function f independent of actual values of x, y unlike the above case of multiplication. Another operation is a max operation:

$$f(x,y) = \max(x,y) \quad \rightarrow \quad \frac{\partial f}{\partial x} = 1(x >= y) \quad \frac{\partial f}{\partial y} = 1(y >= x) \tag{4}$$

That is the gradient is 1 on the input which is larger and 0 on the other input. If $x = 4, y = 2$, then max is x and the function is not sensitive to value of y. If the value of y is increased by a tiny amount h, the function will keep outputting value of x which is 4, hence the gradient is 0. If the value of y is changed by a large amount (in this case larger than 2) then output of f will change. Derivatives shows nothing about the large changes on the function inputs. They only inform about the infinitely tiny amount of changes on the inputs as indicated in eq. 3.

Backpropagation relies on the above explained rule in a more complex neural network. It computes the gradient of loss function with respect to each weight via chain rule based on the above mentioned examples. The computation is performed one layer at a time starting from last layer till the second layer of network. First layer is not included as it is the input layer. Also redundant calculations are avoided by computing derivative of each layer at a time. In other words, the weights are adjusted every time in such a way that loss is minimized and the model reaches at a checkpoint where adjusted weights are very close to target features, hence to get most accurate predictions.

2.4 Activation Functions

These functions are an extremely important feature of a neural network. These functions decide which neuron (a neuron is noting but a mathematical function) would be activated. This means whether the information received by a neuron is relevant or should it be ignored. An activation function performs a non-linear transformation on the input signal and forward it as an input to the next layer of neurons.

Without activation functions weights and bias would be simply doing a linear transformation and a linear equation is easy to solve but very limited to the capacity of solving complex problems. Therefore, a neural network without having an activation function is nothing but a linear regression model which will not be capable of learning or performing complex tasks. Image classification or object detection is a complicated task and would require non-linear transformations. These functions make the process of back propagation possible because of the receiving gradients and error which are a measure to update weights and biases.

2.5 Loss Function

It determines the amount of deviation from the groundtruth or labeled data of the algorithm. Higher loss means the actual outcome is very different than expected result. A high loss function indicates poor performance of the model. There are different ways to define a loss function.

2.6 Neural Network Overview

In supervised machine learning object detection task can be achieved using different approaches. There are various architectures based upon statistical functions. A neural network is made up of neurons having learnable weights and biases. It means that during training process, weights and biases can be updated. A neuron in an artificial neural network is referred to a mathematical function. It receives inputs which can also be output of neurons from a previous layer, weighs each input and sum them up. A weight shows the strength of connection of one neuron to another neuron in next layer. Every neuron has a bias and it determines if the neuron is activated or not. Biases are added to the product of weight and value of neuron. Each neuron is fully connected to all the neurons of previous layer. These neurons do not share any connection within a single layer. The last fully connected layer is referred to as "output layer". The network takes an input and computes the output by taking a dot product. The entire network takes raw image pixels and presents an output in the form of a digit representing the class score for an object. Output layer represents the class scores. It expresses a single differentiable score function. There could be few to many hidden layers and a fully connected layer that has a loss function i.e support vector machine (SVM) or softmax. A typical neural network architecture is shown below:

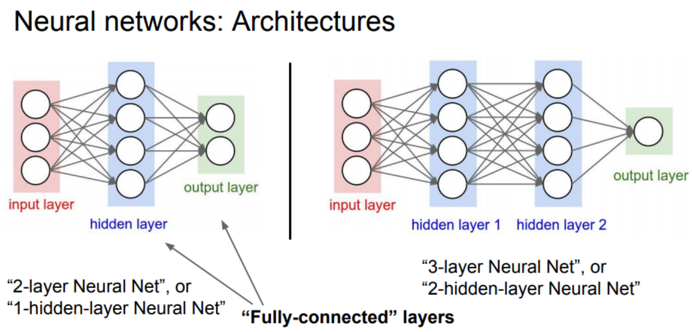


Figure 2: A typical Neural Network architecture

Artificial Neural Networks are very similar to convNets but they do not convolute on an image through a filter thus leading to many more parameters as compared to CNN i.e. in CIFAR-10 dataset, images are of 32x32x3 (width, height, 3 color channels). That means in an input image of size 32x32x3, an Artificial Neural Network would have 3072 weights. These weights tend to increase with size of image.

2.7 Convolutional Neural Network Overview

Just like Artificial Neural Networks as shown in Fig 2, convNets are also made up of neurons consisting of weights and biases which are learnable. Each neuron receives an input then performs a dot product and optionally follows it with a non-linearity. The entire network represents a single differentiable score from the raw image input to the output score. Last fully connected layer has a loss function which represents the error score. This entire process is referred to as forward pass. In convNet, the architecture assume inputs as images which provides flexibility of encoding certain properties. This greatly reduce the amount of parameters to learn and make forward pass more efficient. The layers of convNets have neurons arranged in 3 dimensions (width, height, depth). The neurons in a layer are not connected to all of the neurons in a previous layer like in an Artificial Neural Networks, instead they are connected to only small region of previous layer. The output results into a single vector of class scores by reducing the full image. Following are the types of layers in a Convolutional Neural Network:

2.7.1 Convolutional layer

Lets suppose there is a 2D input image of size 6x6. A filter convolves through the image to extract an output image with desired features. The filter (which is also a matrix) could be of size lets say 3x3, extracts certain features of the image. This process is known as convolutional operation because filter convolve through the image. In example, it would look as shown in fig: 3

INPUT IMAGE

18	54	51	239	244	188
55	121	75	78	95	88
35	24	204	113	109	221
3	154	104	235	25	130
15	253	225	159	78	233
68	85	180	214	245	0

FILTER

1	0	1
0	1	0
1	0	1

OUTPUT IMAGE

429	505	686	856
261	792	412	640
633	653	851	751
608	913	713	657

Figure 3: Convolutional Operation

In fig: 3 the output 429 in 4x4 matrix is obtained by addition of element wise multiplication of the filter with top left 3x3 portion of the input image. Then filter jumps to next pixel and other values are obtained (stride is taken as 1). Output size can be calculated using eq.

$$\frac{n+2p-f}{s}+1 \times \frac{n+2p-f}{s}+1 \tag{5}$$

In the above example n equals 6 as image is 6x6 and f is 3 because of 3x3 filter size, p is padding which equals 0 and s stands for stride which is 1, inserting values in this equation would give us the size of output which is a 4x4 matrix.

2.7.2 Pooling layer

An input image could be large which increases the amount of parameters and introducing a pooling layer helps to reduce the number of parameters. Most commonly used type of pooling is max pooling. Fig 4 shows an example of max pooling with stride and filter size of 2. The idea is to keep the high values in each quadrant because the highest number represents a particular feature and in the example shown in Fig 4, number 6 is the highest value in this quadrant. It means that the most activated pixel in this quadrant is 6 and same goes for other quadrants. The high values are preserved and lower ones are dropped out which are not as activated. Another pooling layer type is average pooling where averaged output of all the pixels is preserved. As it can be seen in the example below that pooling has reduced a 4x4 matrix to just 2x2 matrix, significant amount of parameters are reduced, in addition pooling may also help in reducing overfitting. The resultant matrix after a pooling operation can be obtained from the eq.(1)

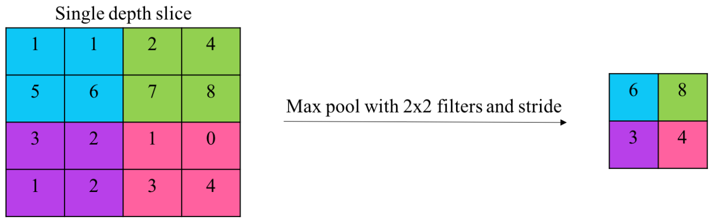


Figure 4: Pooling Operation

2.7.3 Fully Connected layer

After multiple convolution and pooling operations, finally an output can be generated in the form of a class. Convolution and pooling layers only extract the features and reduce amount of parameters from the input images. Fully connected layer (FC layer) is applied at the end to generate the required output equal to the number of classes. There can be multiple FC layers to further minimize the amount of parameters. In convolution the resultant is generation of 3D activation maps whereas the intention is to know whether the image belongs to a particular class or not. The output layer (which generates the class scores) has a loss function and once the forward pass is completed, backpropagation begins to update biases and weights for loss and error reduction. An overview of the architecture is shown below:

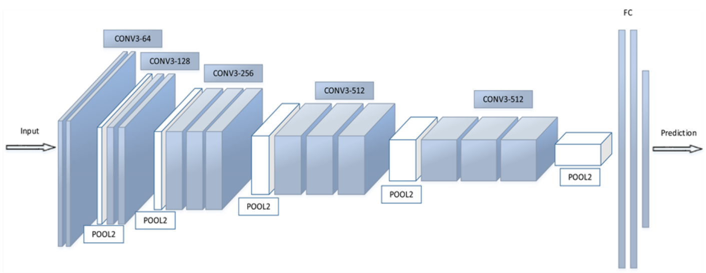


Figure 5: Convolutional Neural Network Architecture [El Khiyari and Wechsler, 2016]

2.8 Binarization Methods

Several binarization methods are applied to experiment best suited thresholding for probability maps. These methods are discussed below.

2.8.1 Otsu's Method

This method was presented by Scholar Otsu. It is widely used till today because of its simplicity and effectiveness. Otsu's method determines a threshold value automatically. In lunar images probability map, the histogram has two peaks. Otsu in simple words take the approximate middle value of the two peaks. This is unlike a global thresholding where an arbitrary value is normally chosen. This method works on the principle of minimizing the intra-class variance, defined as weighted sum of variances of two classes. Given in Figure: 19 is the outcome of this method.

$$\sigma_w^2(t) = \omega_0(t)\sigma_0^2(t) + \omega_1(t)\sigma_1^2(t) \tag{6}$$

Here ω_0 and ω_1 are the probabilities of two classes which are separated by a threshold t. σ_0^2 and σ_1^2 represents the variances of two classes.

2.8.2 Sauvola's Algorithm

It takes graysscale images as input. If an image has three color channels then its necessary to convert that into grayscale image. Savoula proposed to compute a threshold of a grayscale image at each pixel using:

$$T = m \times \left[1 + k \times \left(\frac{s}{R} - 1\right)\right] \tag{7}$$

Here k is user defined parameter, m is mean and s is the local standard deviation computed in a certain window size which is centered on current pixel and R is the dynamic range of standard deviation. $R = 128$ with 8-bit gray scale images. Window size for computation of m and s is user defined. This method is computationally efficient and relatively works well on noisy and blurred documents [Szegedy et al.,]. This method performs over-segmentation which can be visualized in Figure: 24.

2.8.3 Niblack Method

This method was proposed in 1986 by Niblack. Before introduction to this algorithm, global methods were used for image segmentation which are not capable to preserve the minute details of an image during segmentatio. Therefore this method was developed to preserve minute details at local while object separation. A new concept of local window was introduced. Niblack computes the threshold by calculating local mean and standard deviation of pixels value in a local window confined to an image. The equation is given as:

$$T_d = m(x, y) + k \times s(x, y) \tag{8}$$

where $m(x, y)$ is local mean and $s(x, y)$ is standard deviation and k is an image dependent parameter selected manually by the user (normally -0.2 for dark foreground and +0.2 for dark background). The resulting segmentation is not very different from Sauvola's method as can be viewed in Figure: 23.

2.8.4 Adaptive Mean Thresholding

In this type of thresholding, algorithm computes the threshold for a pixel on the basis of a small region around it. This results into various thresholds for different regions of the same image. This gives better results for images which are illumination variant. Outcome of this method is given in Figure: 25.

2.8.5 Mean Thresholding

This threshold is simply mean of the grayscale values of an image. All the values higher than this mean (which is a float number) are considered to be foreground and rest is background. Output of this method can be seen in Figure: ??.

2.8.6 Yen Thresholding

This threshold method takes number of bins used to calculate histogram and returns a threshold based on maximum correlation criterion. It was proposed by [Yen et al., 1995]. Result can be seen in Figure: 21.

2.8.7 Li Thresholding

This method is based on minimum cross entropy. The idea is to select a threshold that minimizes cross entropy between thresholded and original image. It was presented by [Li and Lee, 1993]. Thresholding result is shown in Figure: 26.

2.8.8 Isodata Thresholding

Histogram based threshold alo known as inter-means or Ridler-Calvard method. Threshold is computed by separating the image into two groups of pixels. The resultant threshold lies in the midway between mean intensities of these two groups. That is average of the two. It was proposed by [Ridler et al., 1978]. Figure: 22 shows the outcome of this method.

2.9 Lunar Production Function

Moon provides an ideal test site for studying crater records, particularly since almost all of the lunar endogenic activities ended more than around 3 Giga years (G.y.) with some exceptions [Hiesinger et al., 2000]. Therefore, in last 3 G.y. crater impacts have dominated to change lunar landscape. Space missions have also studied the Moon extensively and collected samples from Moon have provided a unique opportunity to assign age to craters and areas where accumulated crater are counted [Stöffler and Ryder, 2001]. Therefore, on the Moon it an be estimated that cratering rate is the number of craters of a given diameter that are accumulated at given surface during a given time interval.

Crater population is changed by obliteration processes. If it is assumed that a planetary surface is obliterated by some process and crater population starts to develop then crater SFD represents the production SFD of the projectiles. Many authors have tabulated and generalized large amount of crater-counts data in an attempt to understand this production function. Some of the most commonly known lunar crater SFD are proposed by W. Hartmann and G. Neukum.

2.9.1 Hartmann Production Function (HPF)

Hartmann uses a log-incremental SFD representation with a standard bin size for diameter to represent the crater SFD of terrestrial planets. The obtained results are referred as Hartmann production function or HPF. The number of craters per kilometers squared are calculated for a certain diameter range which is $D_L < D < D_R$, this range represents a bin where D_L and D_R are the left and right boundaries of diameter range respectively. The standard bin width is $D_R/D_L = 2^{1/2}$. HPF provides a resultant tabulated data for one specific moment of time, this is the average time of lunar mare surface formation. Findings from most of the lunar mare basalt samples suggests a narrow range of ages between 3.2 to 3.5 G.y. [Stöffler and Ryder, 2001] therefore, the condition of having a fresh surface is satisfied. Some lava flows on the surface could be younger [Hiesinger et al., 2000] therefore the age variation is represented by a factor of 1.1.

The tabulated HPF is considered reliable for the projectile of a production function because the crater counts from different areas of the Moon are combined and averaged. The incremental form of HPF takes form of a piece-wise three segment power law [Ivanov, 2002].

$$\log N_{2^{1/2}} = -2.616 - 3.82 \log D_L, (D < 1.41km) \tag{9}$$

$$\log N_{2^{1/2}} = -2.920 - 1.80 \log D_L, (1.41km < D < 64km) \tag{10}$$

$$\log N_{2^{1/2}} = -2.616 - 3.82 \log D_L, (D > 64km) \tag{11}$$

The function is represented in Figure: 6. Hatmann chose power law segments in 1960s when this work started. Some of the selections were on the basis of historical reasoning that only the craters branch with diameter range between 1.41km and 64km was well established. At that time there were already existing laws of meteorites and asteroids and Hartmann's attempt was to relate those laws to lunar data.

2.9.2 Neukum Production Function (NPF)

Neukum proposed an analytical function describing the cumulative SFD of lunar impact craters. He wrote a series of publications in description of his function. For summaries, see [Neukum and Ivanov, 1994] and [Neukum, 1983]. Neukum showed that the production function is stable since 4 G.y. The time Neukum proposed this function, a full size crater spectrum was known. His approach was different from Hartmann in a way that he computed a polynomial fit to the cumulative number of craters, N per squared kilometers with diameters greater than the provided values of D. Where as Hartmann proposed a piece-wise exponential equations for his production function. For the time period of 1 G.y., Neukum's production function can be represented as

$$\log_{10}(N) = a_0 + \sum_{n=1}^{11} a_n [\log_{10}(D)]^n \tag{12}$$

In above equation D is in km, N is the number of craters with diameters greater than D per squared kilometers per Giga year and values of coefficients a_n are provided in table. The above equation is valid for crater diameters from 0.01km to 300km.

On the basis of age assumption NPF was fit to the crater count. It is notable that both HPF and NPF are a good match for the crater diameter (D) data under 1km range. However, with D > 1km, HPF is much higher than NPF and both functions meet again at diameter of approx. 40km. In Figure: 6 it can be seen that the maximum variation between the two functions is a factor 3 around the diameter of approx. 6km. Note that below the diameter of 1km and between 30-100km, both of the functions are same. After 100km both started to decline but HPF declines more rapid as compared to NPF.

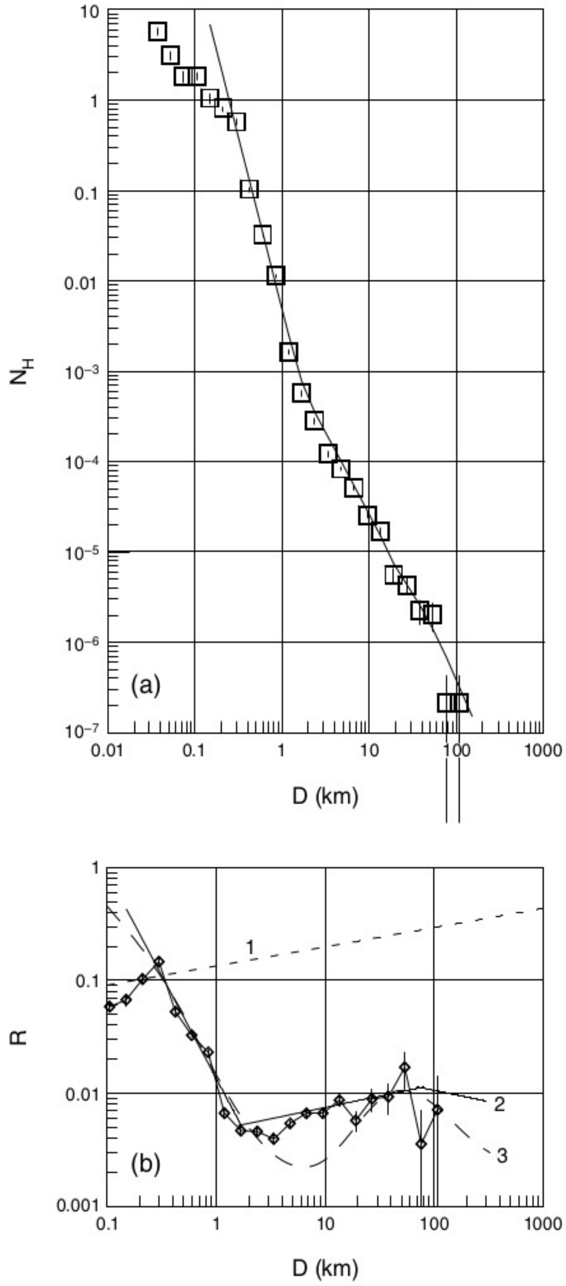


Figure 6: Figure on top shows the representation of the Hartmann production function (HPF). HPF is a function composed of set of points shown in the plot. Straight lines represent the piece-wise power law fitting to the data (equation (15)). Lower figure shows the comparison of Neukum (NPF) and Hartmann (HPF) in the R plot representation. The maximum discrepancy between HPF and NPF (roughly a factor of 3) is observed in the diameter bins around D close to 6 km. D less than 1km and diameter range between 30-100km, both production functions outputs similar results. Fitting the HPF to equation (15), the age estimation is 3.4 G.y. The dashed line 1 represents the approximate saturation level estimated by Hartmann (1995).

3 Related Work

Before wide use of deep learning approaches, different machine learning methods were developed to perform the tasks of object detection. [Viola et al., 2001] proposed a method which was composed of three key components. Firstly objective was to introduce a different image representation called Integral Image to compute rectangle features of the object. The second was an algorithm based on AdaBoost with a purpose to select critical visual features from large amount of data to get a set of classifiers. Third contribution was to combine those classifiers in a cascade to discard the background regions and only compute object-like regions. This algorithm used Haar basis features and was developed to detect faces.

Human detection methods were reviewed by [Dalal and Triggs, 2005] and they came up with Histograms of Oriented Gradient (HOG) descriptors which experimentally outperformed then existing feature sets including wavelets for human detection. A linear support vector machine or (SVM) was used as a baseline classifier. Test was conducted on MIT pedestrian dataset with mostly upright pose. A more challenging set of 1800 images with different poses and background was introduced to test the performance of algorithm and it was concluded that using locally normalized HOG features in a dense overlapping grid provides better results than Haar-like feature approach for human detection.

An algorithm for training support vector machine with a latent structure was proposed by [Felzenszwalb et al., 2008]. This method learn the relationships between HOG features of object parts via a latent SVM which is a semi-convex training problem but once latent information is specified for positive examples then it turns into a convex training. This approach was tested on PASCAL dataset and ranked first in 10 out of 20 classes that entered in PASCAL VOC 2007 competition.

For a long time before identification of objects they were sought to be delineated. This led to rise of segmentation which targets unique partitioning of an image through generic algorithm such that there is one part for object silhouettes in an image [Uijlings et al., 2013]. A selective search methodology was introduced which combines segmentation and exhaustive search. Same as segmentation, image structure was used to guide the sampling process and similar to extensive search the objective was to detect all possible locations of the object. The author introduced a variety of complementary image partitionings to deal with as many image conditions as possible. The resultant search method yielded 99% recall and a Mean Average overlap of 0.879 over 10,000 locations in the test dataset. Results of this methodology were also evaluated on PASCAL VOC 2012 and PASCAL VOC 2010 with mean average precision (mAP) of 0.350 and 0.351 respectively.

Above mentioned algorithms had a noticeable performance but such machine learning techniques had a major requirement which was an involvement of an expert to extract features. Moreover state of the art machine learning techniques also requires a problem to be broken down into different parts and then their results to be combined together at final stage i.e. in SVM a bouding box detection algorithm is needed first to identify all the objects to have histogram of oriented gradients as input to the algorithm that will learn to recognize target objects. These limitations gave rise to deep learning algorithms which are capable of not only handling large amount of data unlike state of the art machine learning techniques but also provide an end to end solution.

Availability of larger labeled datset and competitive GPUs made it possible to implement a deep learning network and test its performance. A notable development in this field took place when [Krizhevsky et al., 2012] showed that a large deep convolutional neural network was able to classify 1.2 million high-resolution images in contest of ImageNet LSVRC-2010. This network was trained into 1000 different classes. On test data top-1 and top-5 error rates of 37.5% and 17.0% were achieved which was better than previous state of the art [Krizhevsky et al., 2012]. This neural network had 60 million parameters with 65,000 neurons and it consisted of five convolutional layers, some of which are followed by max-pooling and three fully connected layers with a final 1000-way softmax activation. A 1000-way softmax because of 1000 classes. Non-saturating neurons was used to optimize training time and to reduce overfitting in fully connecting layers drop-out regularization was used, a newly developed method at that time. A variant of this model was also introduced in ILSVRC-2012 competition and a winning top-5 test error rate of 15.3% was achieved compared to 26.2% by the second-best entry.

The best performing methods were complex assembled systems that typically combine low-level image features with a high-level context [Girshick et al.,]. In 2014 an approach presented by [Girshick et al.,] aims to get better performance on semantic segmentation as well as object detection than other networks at that time. The network is called R-CNN by its authors because it combines regions with CNN features. Therefore R-CNN means Regions with ConvNet features. Detection and segmentation requires localization of objects within an image unlike image classification. This proposed scalable detection algorithm achieved mean average precision mAP of 53.3% on PASCAL VOC 2012. R-CNN was also compared to OverFeat network (a sliding window detector based on a similar CNN architecture) and it was determined that R-CNN outperforms OverFeat by a margin of 7% mAP on 200-class ILSVRC2013. OverFeat network had the best result with 24.3% mAP before introduction of R-CNN (mAP of 31.4%).

R-CNN finds a region of interest (RoI) from an image, creating a warped image region for all the RoIs and forward it to convolutional network. Once each of the region is forwarded, bounding box regressors are applied and classification is done by SVM. These processes are done in three separate models in R-CNN. This resulted in slow training. Fast R-CNN introduces several innovations to improve training and testing speed while also improving detection accuracy [Girshick, 2015]. Fast R-CNN takes in an entire image and forwards it to convolutional network to create a feature map. Then it determines RoI and on top of that it applies a single layer of RoI max pooling followed by a fully connected layer. Then softmax classifers and bounding box regressors are applied. This procedure makes the layer below RoI max pooling trainable which makes Fast R-CNN training a single-stage, using multi-task loss, it also does not require any disk storage for feature caching. Unlike R-CNN, Fast R-CNN uses a single model for feature extraction from regions, dividing them into different classes and returns boundary boxes simultaneously. Fast R-CNN trains very deep VGG16 network 9 times faster than R-CNN and 213 times faster at test-time [Girshick, 2015]. Fast R-CNN achieved 66.1% mAP on PASCAL VOC 2012.

Fast R-CNN used selective search as a proposal to extract RoI which also had a room for optimization. That lead to further development of Fast R-CNN. The next version of this network is called Faster R-CNN. It uses region proposal network (RPN) which takes feature maps of an image as input and generates a set of object proposals and each one of them with an objectness score as output simultaneously. RPNs are trained end to end to generate RoI which are used by Fast R-CNN for detection [Ren et al., 2015]. An alternating optimization was introduced so that RPN and Fast R-CNN can be trained to share convolutional features. Faster R-CNN achieved 70.4% mAP on PASCAL VOC 2012.

Now that it was possible to locate different objects with bounding boxes, could this be extended to locate exact pixels of each object? This problem is known as instance segmentation and was addressed by the framework introduced by [He et al.,]. This method is called Mask R-CNN which extends Faster R-CNN by adding a branch of predicting an object mask in parallel along with existing branch for bounding box recognition [He et al.,]. This branch takes input as CNN feature map and outputs matrix with 1s and 0s. 1 if a pixel belongs to the object and 0 otherwise. This output is known as binary mask as it only has 1s and 0s. Mask R-CNN adds only a small overhead to Faster R-CNN running at 5 frames per second. It does not loose detection accuracy and has been widely used in computer vision for instance segmentation tasks.

Crater detection and categorization process through data-mining from large scale scientific image database was proposed by [Honda and Azuma, 2000]. The detection module was based on state of the art image processing method including binarization, circular object detection using Genetic Algorithm and Hough transformation. Their method took normalized image vectors, discrete cosine transform (DCT) components and intensity histograms as input vectors.

One of the method suggested by [Leroy et al., 2001] was to detect craters using a multi-scale approach based on voting, and tensors as a representation. This method infer curvature estimation from noisy sparse data. The idea was to obtain the oriented normals of the edge curves by applying this method on edge images. This system was applied on Phobos to compute a dense saliency map corresponding to the position and shape of the craters.

Crater densities scaling and impact rates with crater size is an issues which could be addressed by automating crater counting [Vinogradova et al., 2002]. Geological feature cataloging could be performed by labeling images manually but only for limited number of features, handling massive datasets and high resolution images i.e. Mars Global Surveyor, it is required to automate feature identification [Vinogradova et al., 2002]. The Continuously Scalable Template Matching (CSTM) algorithm is an image-matching algorithm used by authors in this paper which was able to detect 88% craters with no false alarms.

Various machine learning algorithms were applied for cataloging impact craters including bagging and AdaBoost, SVM and CSTM. It was found that all the approaches, in particular SVM with normalized image patches provides detection and localization performance is substantially superior to boundary based approaches such as Hough transform [Wetzler et al., 2005]. The receiver operating characteristics (ROC) curves of the tested algorithms are shown in Figure: 7

Most of the craters are formed as the result of meteoroid impacts. The relative formation age of local area of planet could be estimated when frequency of size distribution of meteoroids and its time variations are known. [Sawabe et al., 2006] proposed an algorithm that did not depend on cameras, spatial resolution or sun illumination. It also did not require to tune any parameters either. This algorithm was improvement to their own previously proposed algorithm. Their previous method did not include pyramid representation of the image which was included in later method and thus improving accuracy and reducing processing time. The

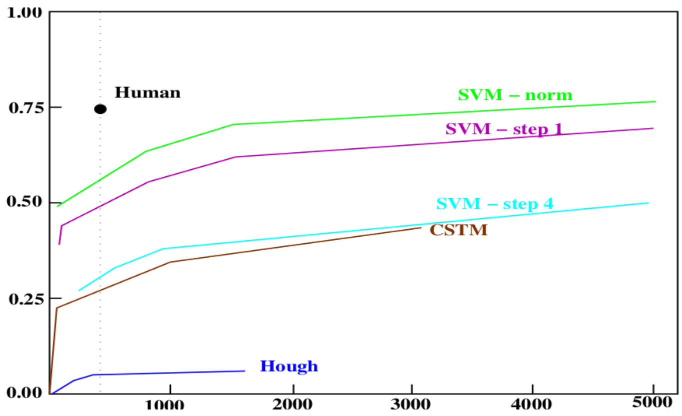


Figure 7: ROC performance of CSTM, SVM and Hough methods. Hough with lowest performance among all other methods [Wetzler et al., 2005].

algorithm was applied on images acquired by Clementine and Apollo under different solar elevation. This approach was able to detect craters with diameter larger than 240m. Accuracy rate was validated by comparing with crater count results of [Neukum et al., 1975] which was 80% extraction of craters with multiple interpreters [Sawabe et al., 2006].

Changeability in appearance of craters and surrounding terrain makes passive imaging based autonomous craters detection difficult to be applied. By experimentation it is proven that unsupervised machine learning methods work well with relatively large craters having clear edge information but their efficiency declines with increase in terrain complexity [Meng et al., 2009].

In 2009 [Martins et al., 2009] performed Viola and Jones (2004) algorithm on Mars dataset gathered by the Mars Orbiter Camera onboard Mars Global Surveyor probe. In this paper author claims that no such method existed that is satisfactory for craters detection. Using this approach craters with diameter larger than 7 pixels were detected. Images of 240m/pixel were used as a data set, hence craters larger than 1680m of diameter were detected. The source of images used are taken from Mars orbiter camera.

An automated system for cataloging impact craters was presented by [Stepinski et al., 2009]. The system used digital elevation model of Mars. The process of crater identification consists of two steps, in first step it identifies round and symmetric topographic depressions as crater candidate and second step identifies crater using a machine learning method. This process is AutoCrad system and applicable to any surface represented by a digital elevation model (DEM) [Stepinski et al., 2009].

A crater detection algorithm (CDA) was presented by [Salamuniccar and Loncaric, 2010] which was based on fuzzy edge detectors and it takes input as digital topography data instead of image data. This algorithm claimed to have more correct detections compared to previous CDA. There were also false positives which were removed manually. The data was taken from Mars Orbiter Laser Altimeter. Most of the work in automating crater detection was performed on DEMs. In 2015 a CDA was proposed by [Emami et al., 2015] which took into account the optical images and the algorithm worked by employing a multi-scale candidate region detection step which was based on convexity cues and candidate region was verified via machine learning. In this paper a CNN classifier was tested against SVM and it was concluded that CNN classifier outperforms SVM both in terms of recall and precision [Emami et al., 2015]. The data used for both training and testing was partially labeled by NASA scientists. Each image consists of 600x400 and taken from LRO. The result of their experiment is shown in Figure: 8.

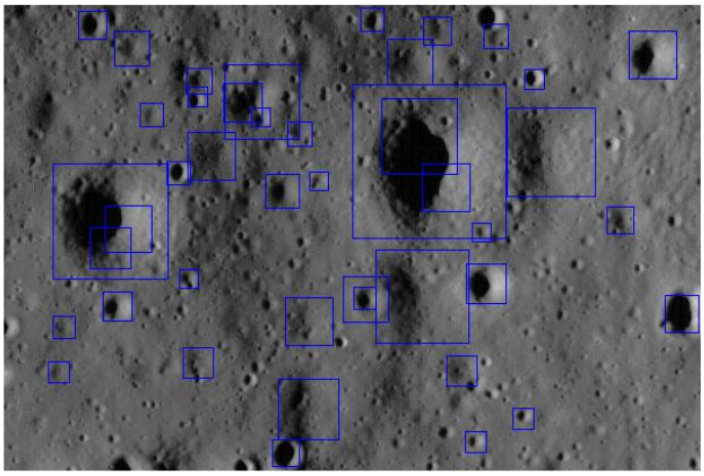


Figure 8: Verified regions in boxes on a test image [Wetzler et al., 2005].

Mars Reconnaissance Orbiter opened a new frontier to automate landforms detection process [Palafox et al., 2017]. Authors came up with two different approaches. These are detection of volcanic rootless cones and transverse aeolian ridges using CNNs and also using SVM with HOG features. They showed that CNNs can detect a wide range of landforms and has a better accuracy and recall than traditional classifiers based on SVMs.

[Wang et al., 2018] presented a method composed of CNN architecture and named it as CraterIDNet which takes remotely sensed planetary images as input and outputs detected craters, their apparent diameters and their positions. The experiments shows that CNN based architecture has the advantages of high robustness, detection and identification accuracy over other methods used by [Urbach and Stepinski, 2009], [Bandeira et al., 2010] and [Ding et al., 2011].

A deep learning model based on CNN architecture was introduced by [Silburt et al., 2019] for crater identification on lunar digital elevation map images also known as DEM. The author also applied transfer learning to craters on Mercury. In this paper random DEM images from LRO and Kaguya were taken as an input data. Thus it was a global gray scale map. The proposed CNN detects only half of the caters per target. The post-CNN recall is lower at 57% ± 20% but detections for craters less than 15 pixels largely improved the post-CNN test recall to 83% ± 16%. The sample of DEM from [Silburt et al., 2019] is shown in Figure: 9.

4 Model Architecture

4.0.1 Sigmoid Function

It is defines as:

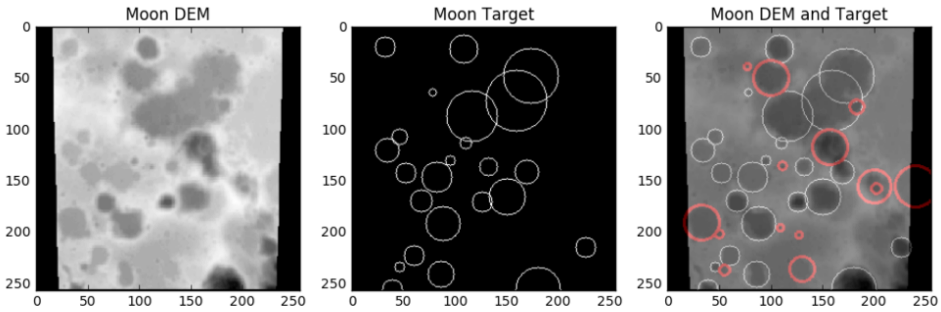


Figure 9: Most left is the Moon DEM sample and middle image shows prediction of craters and most right one shows the missing classifications which are marked in red circles [Silburt et al., 2019].

$$a = \sigma(z) = \frac{1}{1 + e^{-z}} = \frac{e^z}{e^z + 1} \tag{13}$$

where

$$z = Wx_i + b \tag{14}$$

$W being weight vector and x_i is image vector, b stands for bias$

The maximum output of this function is 1 and minimum is 0. Output always lies between values 0 and 1. Plotting a graph of sigmoid function represents it output more clearly:

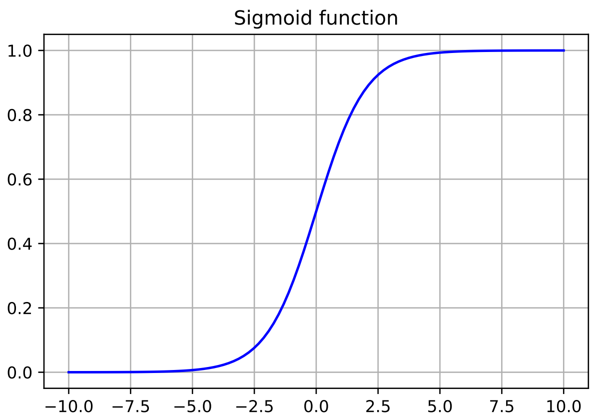


Figure 10: Sigmoid Function

From the graph it can be seen in the graph that z is 0 when curve is passing though 0.5. In convention a rule can be set if i.e. z is greater than 0.5 then output is always 1 and if less than 0.5 then it is 0. A notable behavior of this function is that if z is larger than the dotted region in graph then the derivative of this function is close to zero and same way if it is negative slope is again equal to or close to zero.

This means that it can slow down gradient descent. It also becomes a source of vanishing gradient problem. If the weights are initialized with either very large or very small values then these values saturate the input to sigmoid at very valued region (close to zero or close to one). Even if weights are initialized at a value i.e. 0.2, in a deep network with many layers it will also lead to vanishing gradient problem. In case of only 4 layer network $0.2^4 = 0.0016$ which is small and will get even smaller in next layers. Also the mean of data is 0.5 which meas that data is not centered for the next layer.

It is used for a problem where binary classification is required. In a neural network layer sigmoid must not be used in every layer because of the problems mentioned above. Anyhow, where binary classification is required, it can be useful in the last layer of the network to squash output such as $1 \geq \hat{y} \geq 0$.

4.0.2 ReLU Function

It stands for rectified linear unit and defined as:

$$a = \max(0, z) \tag{15}$$

The graphical form is given as:

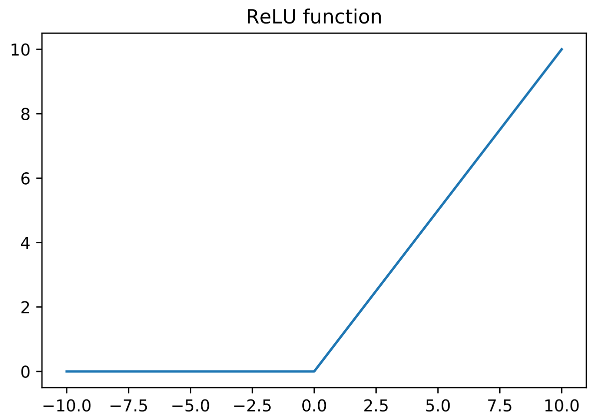


Figure 11: ReLU Function

As seen in the graph ReLU gives an output for a positive value and otherwise the output is 0. It is to be noted that in graph the line looks linear but ReLU is non-linear in nature and it is one of most popular functions used in neural networks because of its simplicity and ability to not let all the neurons fire at once. ReLU is computationally much faster than sigmoid or tanh (written as $a = \tanh(z)$), it does not have exponent computation in such as sigmoid activation function therefore reduces training time significantly in very deep neural networks. [Krizhevsky et al., 2012] observed that training deep CNNs with ReLU is much faster as compared to sigmoid or tanh.

Unlike sigmoid when the receiving input is at the right or left plateau i.e. less than -5 or greater than 5 in Figure: 10 which makes it meaningless to pass a backward pass because of

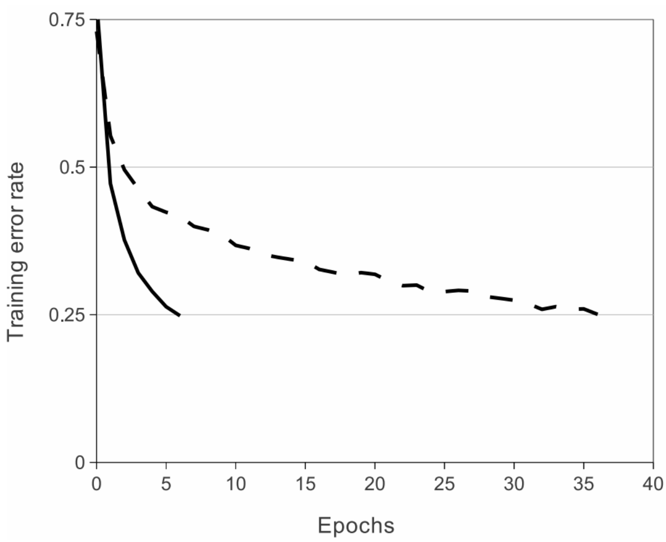


Figure 12: A CNN of four layers trained with ReLU in solid line and tanh in dashed line. It shows that ReLU reaches a 25% training rate on CIFAR-10 six times faster as compared to tanh [Krizhevsky et al., 2012].

derivative being closer to 0, ReLU only saturate when the input is a negative value. Also in Figure: 12 it can be seen that ReLU allows training of larger nets at much less computational costs which means more parameters can be trained at the same computational cost.

4.0.3 Cross Entropy Loss

It is also known as the log loss. It measures the performance of a classification model whose output is a probability value between zero and one. The cross entropy loss increases as the predicted probability diverges from the actual label so predicting a probability of i.e. 0.019 when the actual observation label is one would be bad and it would result in high loss value. Therefore, probability of observation as zero or one (positive or negative) is written as:

$$H(p,q) = -\sum_{x} p(x)\,\log q(x) \tag{16}$$

In the above equation p and q are the cross entropy of true distribution and predicted distribution. The calculation of above equation depends on following factors:

1. Type of layers used in neural network.
2. Type of activation function used. Many activations might not be compatible with calculation because of output value which is either greater than one, negative value or do not sum to one. Therefore, softmax function is often used for multi class classification as it guarantees a well behaved probability distribution function.

A machine learning, a common convention is to represent the ground truth (or labeled) data by vector $\mathbf{y}$ and $\hat{\mathbf{y}}$ is a vector containing the estimate. For a single example the equation is:

$$L = -\mathbf{y} \cdot \log(\hat{\mathbf{y}}) \tag{17}$$

If for example a true label is $\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \end{bmatrix}$ and predictions are $\begin{bmatrix} 0.1 & 0.5 & 0.1 & 0.1 & 0.2 \end{bmatrix}$ then in this specific example all the probability is given to the first value and remaining are zero so they can be ignored. Mathematically it will be calculated as:

$$L = -(1 \times \log(0.1) + 0 \times \log(0.5) + \ldots)$$

$$L = -\log(0.1) \approx 2.303$$

It is clear that loss would be same even if predictions are $\begin{bmatrix} 0.1 & 0.5 & 0.1 & 0.1 & 0.2 \end{bmatrix}$ or $\begin{bmatrix} 0.1 & 0.6 & 0.1 & 0.1 & 0.1 \end{bmatrix}$. This is a key feature of multi class cross entropy loss. The value does not depend on how probability is split between incorrect classes, it only penalizes the probability of correct class.

The model architecture is composed of contracting and expansive paths as shown in the Figure: 14. Like in a typical CNN, the image becomes smaller as a result of convolutional operations. U-net has the same effect and thus left part of this pipeline is referred to as contracting path. It consists of several convolutional operations such that each of two un-padded convolutional layers with an activation of rectified linear unit (ReLU) are followed by a layer of max pooling of the size 2x2. This results into downsampling of the image. During each downsampling step the number feature channels are doubled as seen in the Figure: 14. This part ends up with a dropout function before it starts expanding. Dropout is nothing but regularization which reduces interdependent learning amongst neurons. This reduces the possibility of overfitting the model.

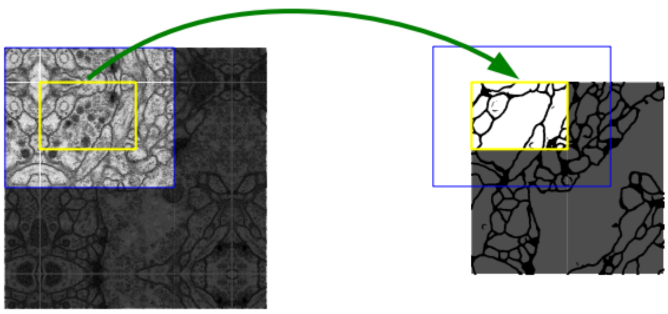


Figure 13: This is example of neuronal structures segmentation. It shows the overlap-tile strategy for seamless segmentation of large images. The yellow area is target for prediction and data inside the blue area is required as in input for prediction. Missing input data is exptrapolated by mirroring [Ronneberger et al., 2015].

The right part of Figure: 14 illustrates the expansive path where every step consists of upsampling of feature map which is followed by a 2x2 convolutional operation (also referred to as up-convolution). This halves the number of feature channels. The gray arrow shows the concatenation of corresponding feature map from contracting path where it is cropped from the image. It is then subjected to two 3x3 convolutions and each followed up by a

ReLU activation function. The cropping is required because of loss of boarder pixels during convolutions. At the final layer a 1x1 convolution maps each 64 component feature vector to the target number of classes i.e. background class and crater class. The final layer follows up by sigmoid activation function which determines the output of a class score between 0 to 1. Finally, binary cross entropy determines the loss while training of algorithm. In this model the total number of convolutional layers amounts to 23. It is important to select the input image size so that max pooling operations are applied to a layer with the same x and y size. This allows seamless tiling of segmentation map in output as shown in Figure: 13

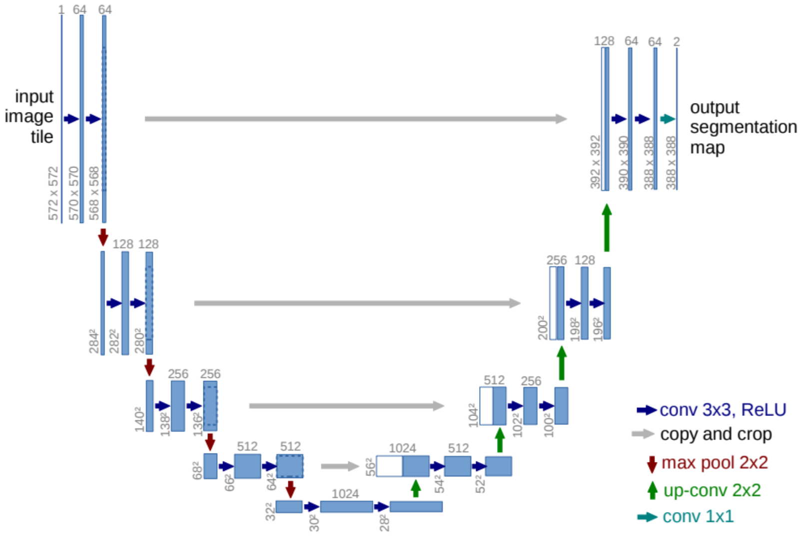


Figure 14: U-net Architecture [Ronneberger et al., 2015].

5 Methodology

This section is divided into following parts:

- Data Set
- Pre-processing
- Training of Network
- Postprocessing to get segmented images

5.1 Data Set

The dataset which is in the form of gray scale lunar images is taken from lunar reconnaissance orbiter camera (LROC) archive *http://wms.lroc.asu.edu/lroc/search*. Images from this data source are very large in size (5064x52224) with resolution of 1.009. Which means processing these images would require a competitive hardware. Also each image in this data has several thousands of craters with sizes ranging from very few meters to several hundred meters. Keeping in view the size of an image and amount of craters it contains, it was considered to take one of the image and crop it into several small images of equal size and width. This dimension was chosen based upon the ease for annotation process. Smaller image size would mean less craters to annotate in a single image and therefore more number of images in training set which would make data handling such as pre-processing and post-processing simpler. The amount of cropped images are 300 each with size 350x350 and are gray scale images.

5.2 Pre-processing

In this step before training of Neural Network images are processed for annotation and extracting labeled masks for the preparation of input data set for training. It involves following step.

- Image Annotation
- Extraction of Binary Masks

5.2.1 Image Annotation

It is one of the most important tasks in computer vision problem related to deep learning. This task is done by human, images are annotated with labels. These are established by the annotator who can perform this task on the basis of simple instructions. No skilled labor is required to do this task. These labels carry information to the Neural Network about what are the target features in image. In this particular project there is only one label which is "crater" and has shape of a circle.

Annotation (labeling) could be laborious and time taking task. It is specially required when a network is not trained on the same type of class as required for predictions. In the case of lunar craters, there is not a great amount of work that has been done before in terms of machine learning and finding already annotated data set seems difficult to find. Therefore labeling was needed for training purpose.

After performing few annotation experiments from the above mentioned tools, VGG Image Annotator was found most simple and faster to annotate lunar crater dataset. This tool stores annotations in the form of a json format. Once the labeling is complete, a large json file represents the craters with a center point as x and y coordinates and radius respectively.

In Figure: 15, images in the first row are part of the training dataset and below them are the annotations. Images marked with circles on craters are not intended to be used for the training of algorithm, those are only visualizations to labeled data so that wrong annotations could be removed or missing ones could be added.

5.2.2 Extraction of Binary Masks

Like mentioned before the result of these annotations (json) is projected to get the binary masks of craters. These masks are binary images that represents the black and white pixel values of images. White pixels means craters and black means background class as shown in the Figure: 16. White circles are craters belonging to Figure: 15. These provide learning

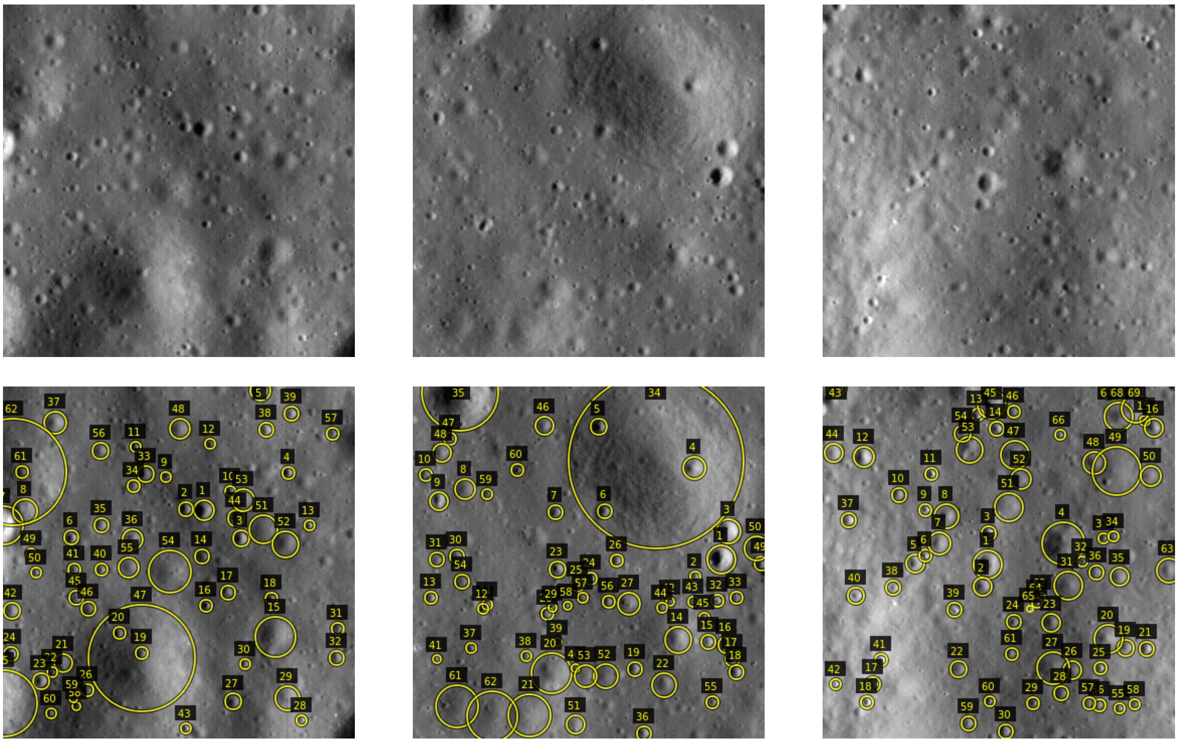


Figure 15: Example of cropped images and corresponding labeled images. Annotation is performed using VIA tool

objective to Neural Network that how the shape looks like and what features to learn. These binary masks have pixel values either 0 or 255 and are of the same shape along x and y as of the images. These masks are normalized in side the algorithm architecture to values 0 and 1.

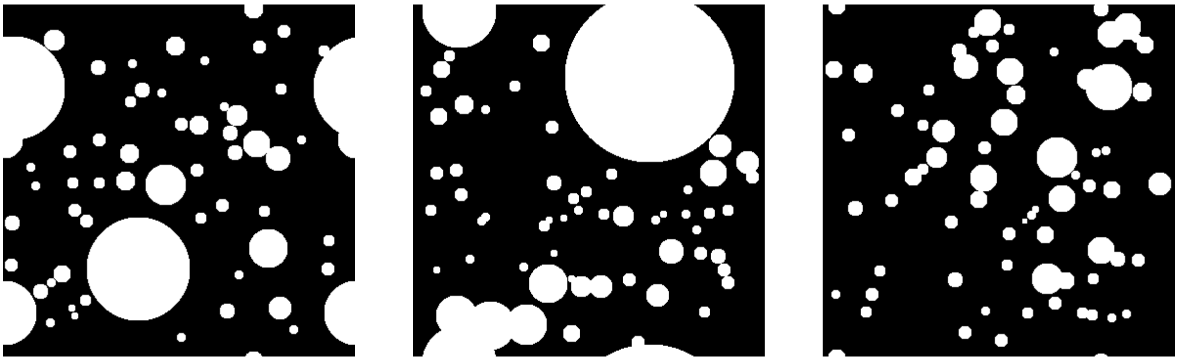


Figure 16: Visualization of binary masks after projection from annotated (json format) images that belong to Figure: 15

5.2.3 Data Augmentation

To create larger dataset without additional annotations, data augmentation is necessary. Creation of a larger dataset is done to achieve better training for the algorithm. Augmentation creates multiple images out of one image which means larger dataset for training of algorithm and thus achieve better performance on test data. Therefore every training image is randomly augmented in following ways:

1. Rotation: Random rotation of images from 0° to 360° angle.
2. Translation: Random change in shift in x and y direction of image between -4 to 4 pixels.
3. Flipping: Image is flipped with a factor of 0.5 as probability.

The images are randomly perturbed during training so that augmentation can have maximum effect. This way model have a low probability of seeing same type of training image more than once. In lunar crater dataset there are no RGB images hence color augmentation is opted out.

5.3 Training

The data composed of images and their respective masks is divided into training (80%) and validation (20%) data set. This percentage practice is typical in machine learning, however it does not have to be always at this rate. In case of large data set i.e. several thousands of images, 60% or even less can be set for training. In this particular case there is not test data taken out from the data set because of limited annotations. As mentioned in introduction, test data is taken from another thesis student who has labeled all the craters manually.

The input images along with their corresponding segmentation maps are utilized to train network with stochastic gradient decent implementation of Keras (originally written in Cafe [Ronneberger et al., 2015]). Whereas Keras implementation can be found here: *https://github.com/zhixuhao/unet*. Inside training pipeline, the convolutions are unpadded which leads to a smaller output size after every layer. This decrease in size is of the factor of constant border width. The batch size is reduced to a single image to minimize the overhead and utilize maximum GPU memory. In this configuration momentum is kept high (0.99) such that a larger number of already seen training samples determine the update in ongoing optimization step. Momentum helps to accelerate gradients vectors in the right direction. A softmax function over the final feature map combined with cross entropy loss function determines the energy function. In softmax function as defined in eq.7, z_j denotes the activation in feature channel j at the pixel position $z \in \Omega$ with $\Omega \subset \mathbb{Z}^2$. K is the number of classes and $f_j z$ is the approximated maximum function. This means that $f_j z$ will be close to 1 for j with maximum activation and vice versa for other values of j. The cross entropy function penalizes each deviation of $q(x)$ from 1 as defined in eq.1.

The weights are saved in a checkpoint when losses are less than previous saved checkpoint. Any of these saved weights could be utilized to apply on a test dataset but definitely saved weights with best performance on validation dataset is the best choice to be deployed on a test dataset.

5.4 Contrast Limited Adaptive Histogram Equalization (CLAHE)

Histogram equilization considers the global contrast of an image. Therefore in cases where image has high pixel values, such regions tend to loose alot of information as those pixels are stretched further towards 255 and object becomes too bright.

By using CLAHE, the image is divided into small blocks also called tiles (function implemented in OpenCV has default tile size of 8x8). Then each one of these tiles histogram equalized. This makes histogram confined to a small area. It will be amplified if noise is there. Contrast limiting is applied to avoid this problem. The default contrast limit is 40 in OpenCV and pixels with larger values are clipped and distributed to other bins uniformly before application of histogram equilization. After that there are artifacts in tile boarders which are removed by the application of bilinear interpolation. By applying CLAHE the detection of craters increased up to 40%.

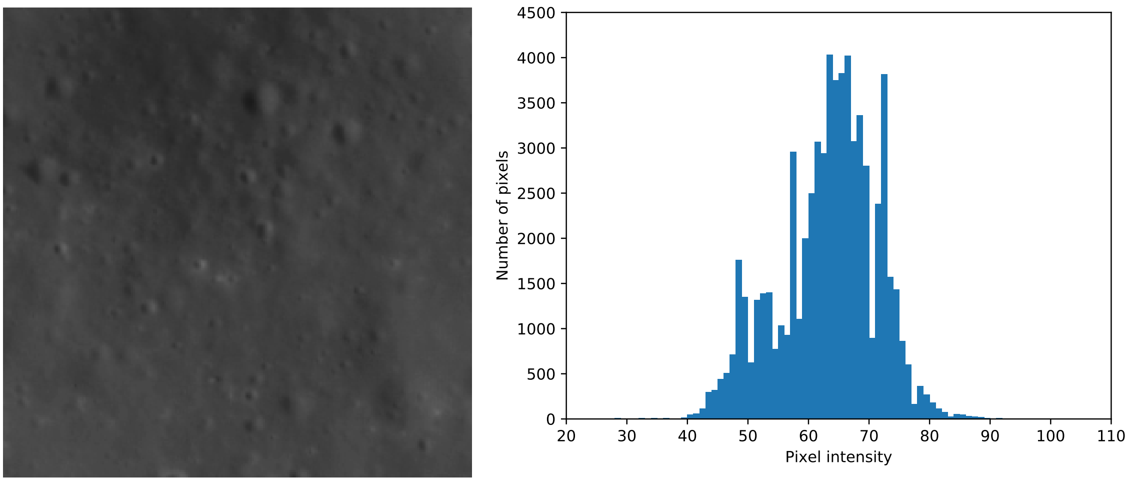


Figure 17: Histogram of an image before contrast limited adaptive histogram equilization

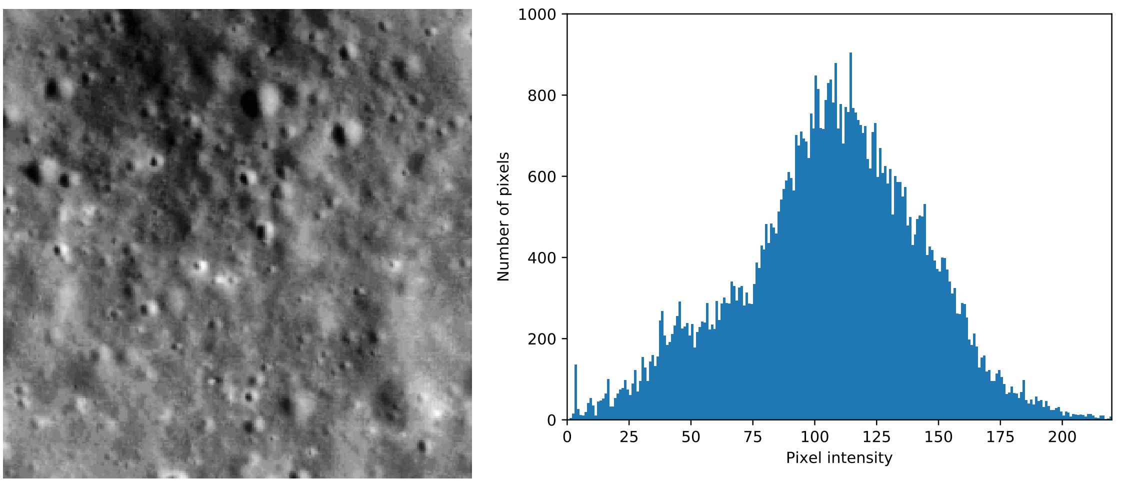


Figure 18: Histogram of an image after contrast limited adaptive histogram equilization

5.5 Prediction

The best trained model is used for prediction on test datset. Predictions yield a probability map of each image in which light pixels depict maximum likelihood and dark pixels represents less probability of crater existence. Such heat map of one of the test images is shown in binarization methods.

5.6 Post-processing

The resulting predictions from CNN model are post processed in which the amount of detected craters are counted along with respective diameters and location by using region proposal algorithm proposed by [Burger et al., 2009] on binarized images by several binarization methods discussed in binarization section. The test images are taken from similar datset to training with same sun illumination angle as well as with different illumination (geographic location of Apollo 17) where number of craters are annotated by [Polegubic, 2020]. The age of this location is already known. Therefore by identifying CSFD and crater diameters it is possible to estimate age which is plotted in a log-log plot.

5.7 Evaluation Method

Precision, recall and F1-score evaluate the performance of algorithm. Precision is defined as the percentage of results which are relevant and mathematically written as:

$$\text{Precision} = \frac{\text{True positive}}{\text{True positive} + \text{False positive}}$$

True positives are correctly identified craters and false positives are craters detected by the algorithm but no crater exists on that location.

Recall is defined as the percentage of total relevant results correctly classified by the algorithm and is given by:

$$\text{Recall} = \frac{\text{True positive}}{\text{True positive} + \text{False negative}}$$

In the above equation false negatives are simply missed craters by the algorithms.

For overall evaluation both of these metrics are taken into account and finally F1-score has been computed by the given equation:

$$\text{F1-score} = 2 * \frac{\text{Precision} \ * \ \text{Recall}}{\text{Precision} \ + \ \text{Recall}}$$

Hence higher the F1-score better is the performance of algorithm and vice versa. Following the experiments of [Emami et al., 2015] a verified region is a true positive if it has more than 40% overlap with a ground truth crater; otherwise, it is a false positive .

6 Experimental Results and Discussion

Trained model is used to predict on two testsets. First one is similar to training dataset but test images are not seen by algorithm during training or evaluation. Second dataset is annotated by [Polegubic, 2020] which has a different sun illumination angle as well as incidence angle of the spacecraft. Number of experiments are performed using several binarization methods at different ranges from their respective threshold values. These results are given in Table: 1.

Apart from region proposal algorithm, Hough circle transformation is also used for circle fitting but overall results are poor. It does a reasonable job of detecting some of large-sized or medium craters but overall performance is quite bad compared to SVM and CSTM [Wetzler et al., 2005]. Hough circle fitting on one of the test images can be visualized in Figure: 28.

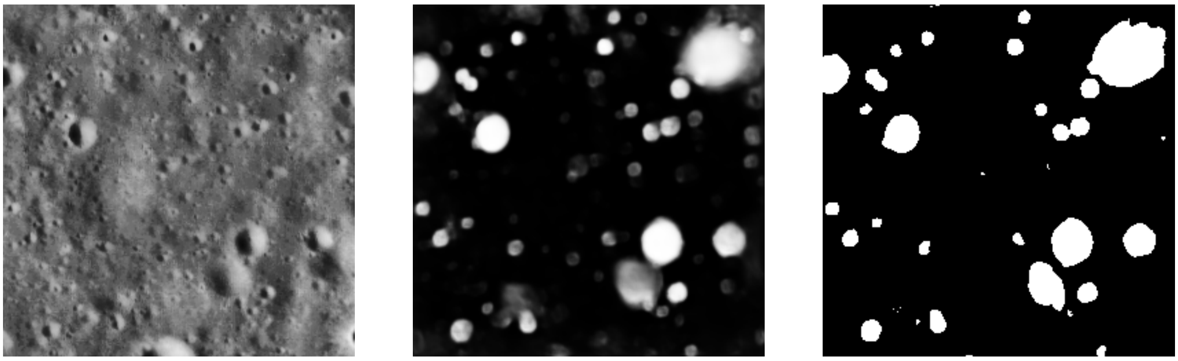


Figure 19: Otsu thresholding

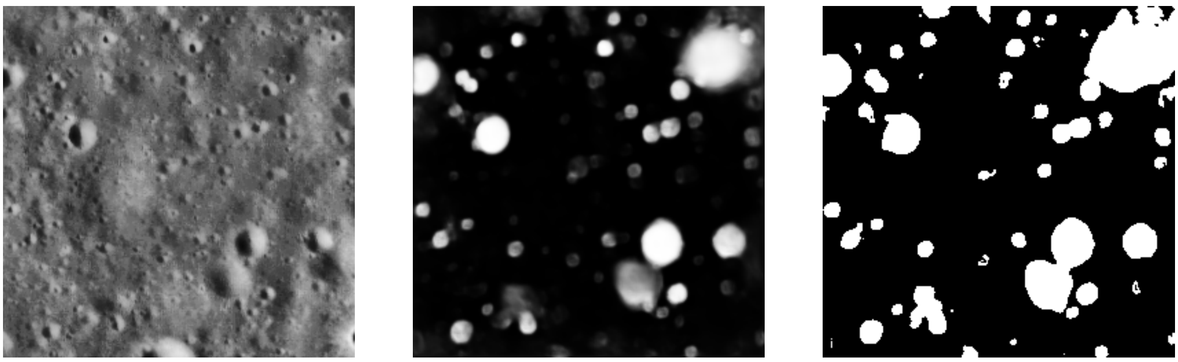


Figure 21: Yen thresholding. Right most image is the thresholded image of the predicted probability mapped image in the middle. Most left is the actual image.

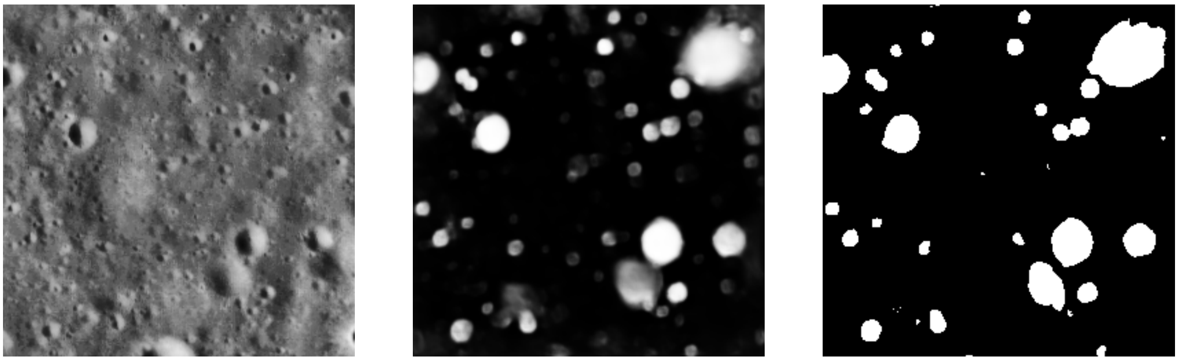


Figure 22: Isodata thresholding. Right most image is the thresholded image of the predicted probability mapped image in the middle. Most left is the actual image.

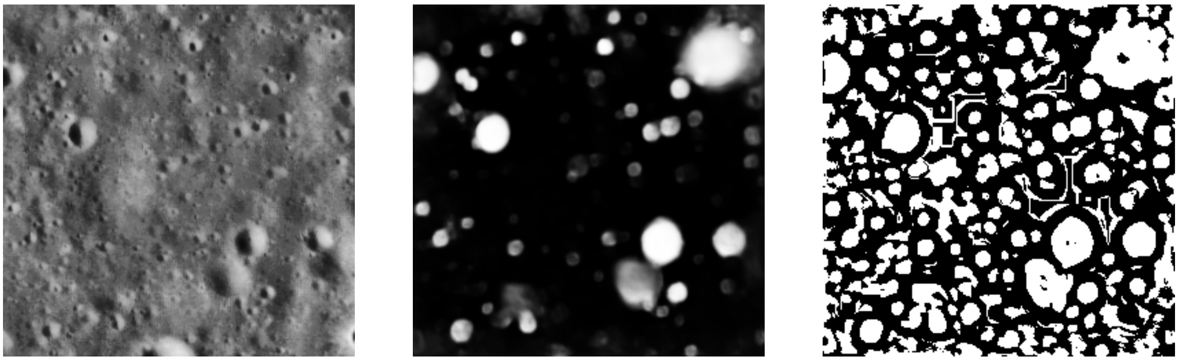


Figure 23: Niblack thresholding. Right most image is the thresholded image of the predicted probability mapped image in the middle. Most left is the actual image.

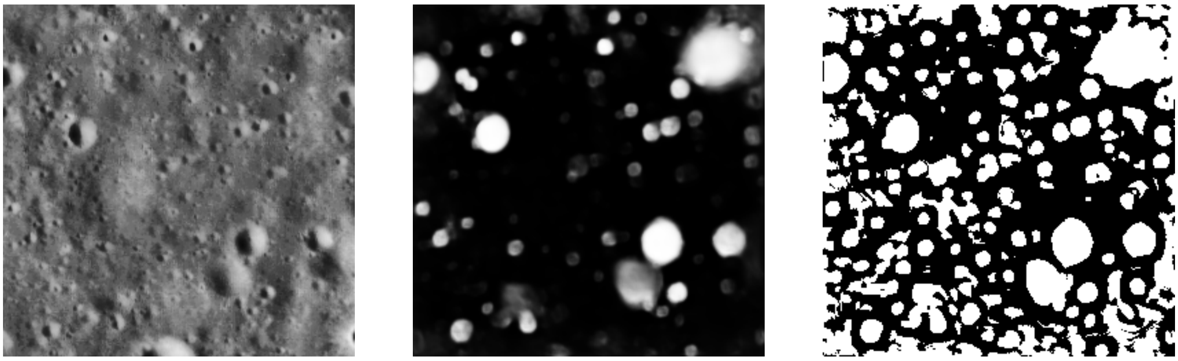


Figure 24: Sauvola thresholding

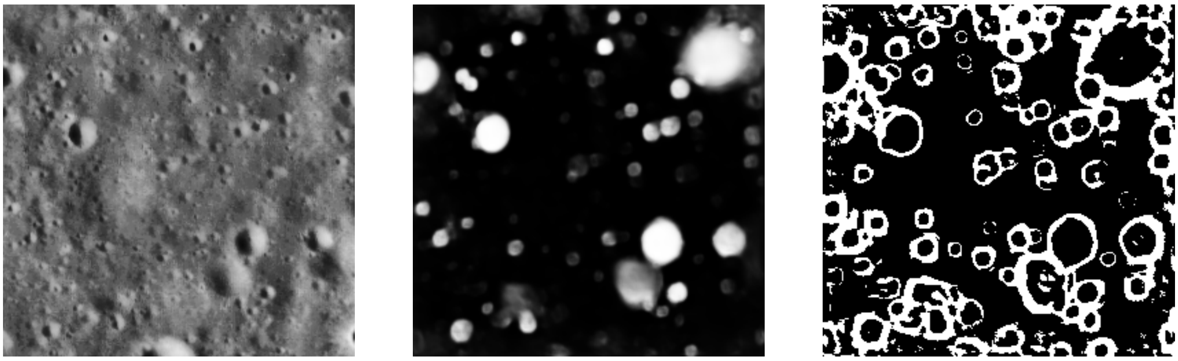


Figure 25: Adaptive mean thresholding. Right most image is the thresholded image of the predicted probability mapped image in the middle. Most left is the actual image.

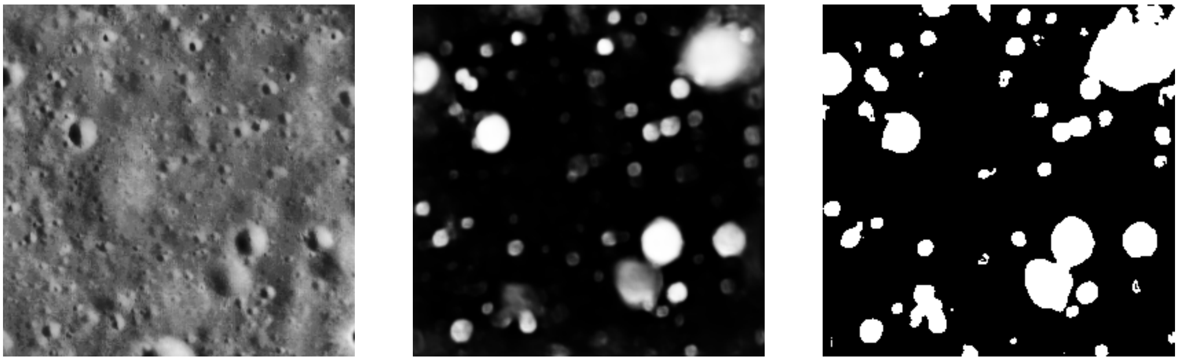


Figure 26: Li thresholding. Right most image is the thresholded image of the predicted probability mapped image in the middle. Most left is the actual image.

Table 1: F1 scores on applied binarization methods on lunar crater probability map. First row shows the percent decrease (indicated with a negative sign) and percent increase of binarization method.

	-50 %	-40 %	-30 %	-20 %	-10 %	0%	10%	20%	30%	40%	50%
Otsu	51.9	53.3	53.0	51.7	52.8	51.9	50.7	50.1	47.9	46.7	45.9
Isodata	47.6	49.2	47.7	48.4	48.6	46.9	47.4	47.6	44.7	44.6	44.1
Yen	36.7	38.7	41.4	44.3	45.1	45.8	45.8	47.7	48.3	49.2	49.4
Li	36.7	38.7	44.2	44.4	45.2	45.8	45.8	48.2	48.4	49.5	49.3
Mean	32.5	35.1	36.0	36.7	38.8	40.1	42.6	44.4	44.6	46.0	45.8
Adaptive mean	2.2										
Adaptive Gaussian	9.3										
Sauvola	14.0										
Niblack	9.3										

Experimental results given in Table: 1 are performed on [Polegubic, 2020] dataset. The dataset consists of a single image of the size 3130x2427 pixels and it is cropped into several

256x256 images for predictions. Cropping is done because the algorithm takes this input size for both training and prediction. Then prediction is performed on all the images and resulting predictions are concatenated together in an order similar to the original image. Then the experiment presented in Table: 1 shows computation of F1-score on several binarization techniques to find out which one yields highest F1-score.

Scores given in Table: 1 are taken as benchmarks for image binarization. It can be seen that the F1-score increases by increasing threshold for methods like Yen, Li and Mean but for Otsu and Isodata it decreases slowly by further increasing threshold. The thresholding methods results into slightly over or under segmentation in the case of crater detection. It can be seen that Otsu and Isodata results into a high threshold value thus by decreasing threshold it is likely that F1-score might increase. Rest of the methods results into over segmentation that specially penalizes recall thus resulting F1-score is lower in comparison. This makes sense and observed in experimentation.

Relationship between precision and recall of different filters is shown in Figure: 27. Adaptive Mean, Adaptive Gausssian, Sauvola and Niblack are not shown in graphs because of their poor performance with respect to F1-scores as shown in Table: 1. It is clear that by setting higher threshold reduces recall but increases precision. This can be observed from Otsu and Isodata methods as both results into higher threshold values. The difference between recall and precision is obvious only if the segmenter is strongly over or under splitting [Badrinarayanan et al., 2017]. In any of the case precision can be misleading as evaluation metric because it favors under segmentation. Contrary to precision, recall is does not favor under or over segmentation, therefore F1-score is taken into account as the evaluation metric for segmentation accuracy. From graphs it can be noted that Otsu and Isodata provides a good trade off between precision and recall thus results into higher F1-score than other methods.

Table 2: Patch wise mean F1 scores on test set distribution from dataset split

	Precision	Recall	F1 score
Otsu	77.5	70.6	73.8
Isodata	76.9	70.1	73.3
Li	56.9	80.5	66.7
Yen	51.8	72.9	60.5

Table 2 shows the F1-score on 20 test images taken out from dataset used for training. Computation is performed on each image separately then average of all the images is taken. Given filters are chosen on the basis of experimentation of different filters and their F1 comparison given in Table 1. It can be seen that Otsu has the best precision because of higher threshold value which yields less false positives. High threshold has a drawback of missing small craters which are usually less than 6 pixels and also those which are highly degraded and barely have a shadow. This can be seen by threshold of Li and Yen's method where recall is high but precision is punished. Overall, Otsu outperforms are the other binarization methods.

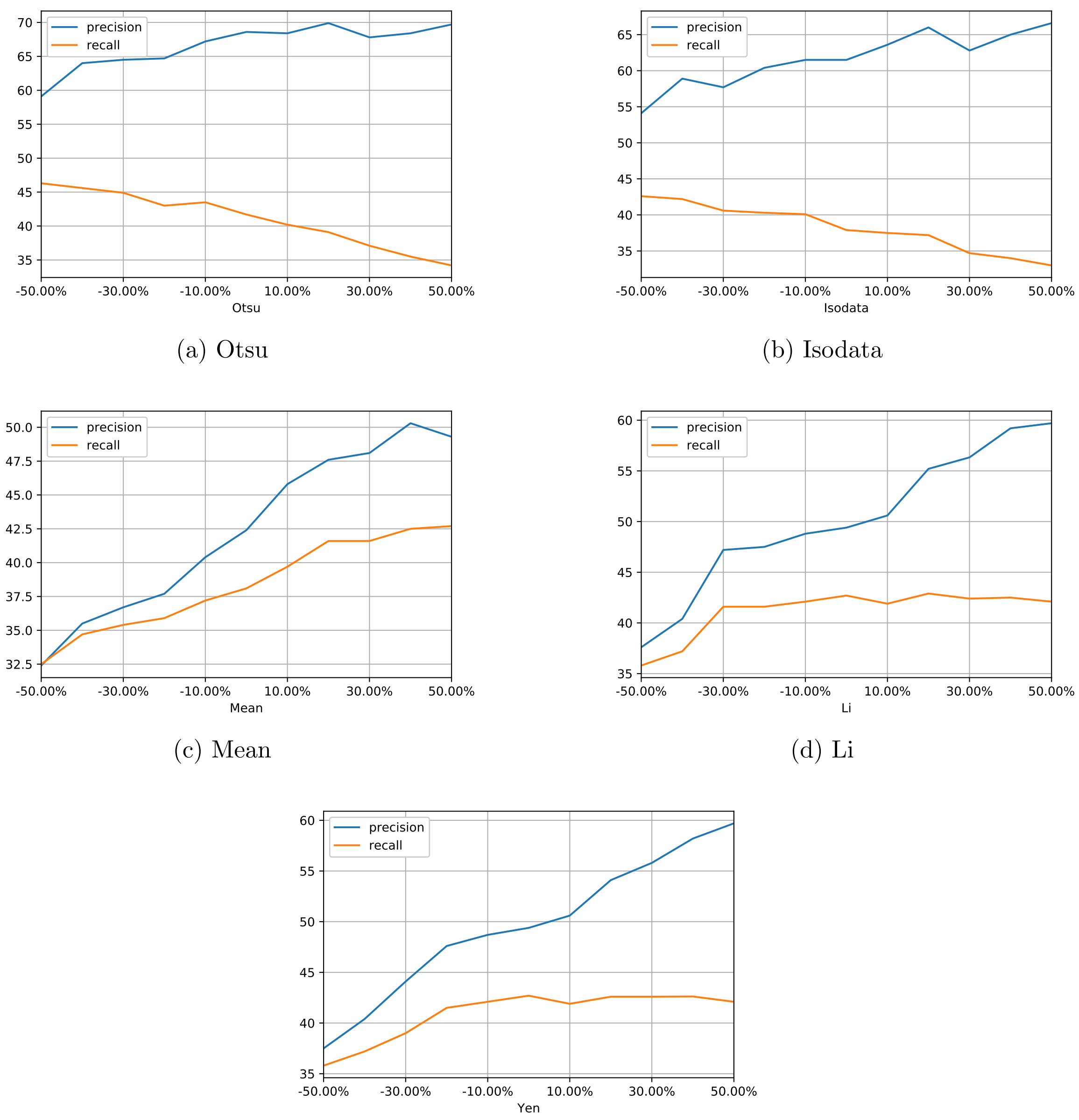
Table 3: Patch wise mean F1 scores on a different test annotated from [Polegubic, 2020]

	Precision	Recall	F1 score
Otsu	77.0	55.7	64.6
Isodata	73.9	53.8	62.3
Li	60.7	59.2	59.9
Yen	51.9	53.1	52.5

Table 3 shows F1 results on [Polegubic, 2020] dataset. It is expected to see low F1-score compared to actual test set because of difference of shade length. Precision has not changed too much but there is notable decrement in recall score of methods and after analysing images visually it is concluded to have happened mainly because of different sun illumination angle. Otsu once again outperforms other methods including Isodata by a slight margin.

It has been observed that taking a larger image than 256x256, cropping it into 256x256, performing predictions and concatenating prediction images back together to form the actual image results into lower F1-score than computation on every image separately. This is due to the fact that many of the detected craters are cut into two different images and some are located in such a way on one image there is shadow and other has sun illumination. This results into detection of crater on one image but not on the other which there is one true positive and one false negative which recall is 50% instead of 100%. That is why scores in Table 1 are much lower than Table 2 and 3.

Mask R-CNN which is a deep neural network based architecture was also applied in experimentation. It is observed that performance of this architecture was poor despite of huge success on several types of object classification and instance segmentation problems. It is believed that this happened because of not enough dataset for sufficient training. Lunar craters does not resemble with the classes it is trained on i.e. COCO dataset. Moreover, the complexity of craters in optical images with shades on one side makes it difficult for such a deep architecture to learn from a dataset composed of less than 300 images. However, further research is needed to draw more concrete conclusions in this regard.



(a) Otsu (b) Isodata

(c) Mean (d) Li

(e) Yen

Figure 27: Relationship between precision and recall on different binarization methods with upto 50% increment and 50% decrement from resultant threshold values at 10% intervals.

Hough circle transformation was also applied for blob detection but the performance is very poor. One of the image is shown with several craters detected by the algorithm but Hough circle method extracted only 4 of them. This phenomena was also observed by [Wetzler et al., 2005] and his performance curves can be seen in Figure: 7.

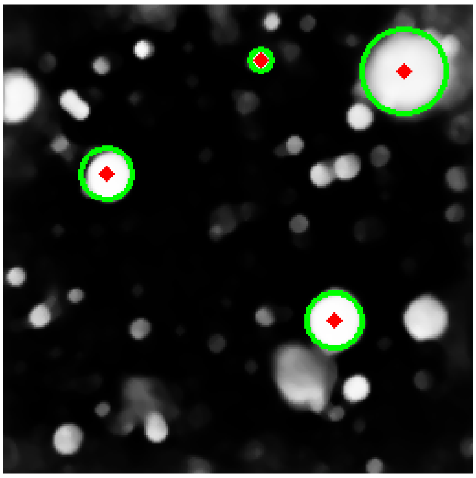


Figure 28: Hough circle fitting on predicted image.

6.1 Plotting Age Curve

Another evaluation is comparison of age with the determined age of crater counted surface by planetary scientists.

7 Conclusion

Crater automation can be challenging because of variability in appearance of craters and surrounding terrain. U-Net based architecture was applied to detect craters in optical images taken by Lunar Reconnaissance Orbiter. It is observed that by applying Contrast Limited Adaptive Histogram Equalization (CLAHE) the detection is increased up to 40%. The resulting probability or heat maps were subjected to several binarization methods to create blobs for the lighter pixels in an image and Regional Proposal algorithm (implemented in scikit) was applied to detect blobs along with their respective diameters. Experiments show that Otsu's binarization method performs best even though Otsu's threshold creates an under-segmentation map.

It is also observed that cropping a large image and performing predictions on smaller cropped images will result into artifacts when segmented maps are patched together. After binarization of patched large image, it results into addition of false positives on reflection to artifacts which penalizes F1-score and has been observed in Table: 1, therefore it makes more sense to perform image wise binarization and calculate F1-score on single image basis.

Several deep learning architectures have been applied to solve crater detection problem but largely are experimented with digital elevation models which eliminates small craters. In this thesis, U-Net was applied on optical images and it was tested on a proportion of dataset used for training and also on a different testset which is annotated by [Polegubic, 2020] with tighter solar longitude angle as compared to training dataset. An F1-score of 73% was achieved on first testset whereas change in solar angle reduced F1-score to a value of 64%.

References

[Badrinarayanan et al., 2017] Badrinarayanan, V., Kendall, A., and Cipolla, R. (2017). Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(12):2481–2495.

[Bandeira et al., 2010] Bandeira, L., Ding, W., and Stepinski, T. (2010). Automatic detection of sub-km craters using shape and texture information. In *Lunar and Planetary Science Conference*, volume 41, page 1144.

[Brunelli and Poggio, 1993] Brunelli, R. and Poggio, T. (1993). Face recognition: Features versus templates. *IEEE transactions on pattern analysis and machine intelligence*, 15(10):1042–1052.

[Burger et al., 2009] Burger, W., Burge, M. J., Burge, M. J., and Burge, M. J. (2009). *Principles of digital image processing*, volume 54. Springer.

[Burl et al., 2001] Burl, M. C., Stough, T., Colwell, W., Bierhaus, E., Merline, W., and Chapman, C. (2001). Automated detection of craters and other geological features.

[Dalal and Triggs, 2005] Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection.

[Ding et al., 2011] Ding, W., Stepinski, T. F., Mu, Y., Bandeira, L., Ricardo, R., Wu, Y., Lu, Z., Cao, T., and Wu, X. (2011). Subkilometer crater discovery with boosting and transfer learning. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(4):39.

[El Khiyari and Wechsler, 2016] El Khiyari, H. and Wechsler, H. (2016). Face recognition across time lapse using convolutional neural networks. *Journal of Information Security*, 7(03):141.

[Emami et al., 2015] Emami, E., Bebis, G., Nefian, A., and Fong, T. (2015). Automatic crater detection using convex grouping and convolutional neural networks. In *International Symposium on Visual Computing*, pages 213–224. Springer.

[Felzenszwalb et al., 2008] Felzenszwalb, P., McAllester, D., and Ramanan, D. (2008). A discriminatively trained, multiscale, deformable part model. In *2008 IEEE conference on computer vision and pattern recognition*, pages 1–8. IEEE.

[Girshick, 2015] Girshick, R. (2015). Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448.

[Girshick et al.,] Girshick, R., Donahue, J., Darrell, T., and Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation.

[Greeley et al., 1970] Greeley et al., Gault, D. E. (1970). Precision size-frequency distributions of craters for 12 selected areas of the lunar surface. *The Moon*, 2(1):10–77.

[Hartmann, 1981] Hartmann, W. K. (1981). Chronology of planetary volcanism by comparative studies of planetary cratering. *Basaltic Volcanism on the Terrestrial Planets.*, pages 1049–1127.

[He et al.,] He, K., Gkioxari, G., Dollár, P., and Girshick, R. Mask r-CNN.

[Hiesinger et al., 2000] Hiesinger, H., Jaumann, R., Neukum, G., and Head III, J. W. (2000). Ages of mare basalts on the lunar nearside. *Journal of Geophysical Research: Planets*, 105(E12):29239–29275.

[Honda and Azuma, 2000] Honda, R. and Azuma, R. (2000). Crater extraction and classification system for lunar images. *Mem. Fac. Sci. Kochi Univ.(Inform. Sci.)*, 21:13–22.

[Ivanov, 2002] Ivanov, B. (2002). The comparison of size-frequency distributions of impact craters and asteroids and the planetary cratering rate. *Asteroids III*, 1:89–101.

[Ivanov et al., 2015] Ivanov, T. I., Huertas, A., and Johnson, A. E. (2015). Probabilistic surface characterization for safe landing hazard detection and avoidance (hda). US Patent 9,141,113.

[Koeberl, 1994] Koeberl, C. (1994). African meteorite impact craters: Characteristics and geological importance. *Journal of African Earth Sciences*, 18(4):263–295.

[Krizhevsky et al., 2012] Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.

[Leroy et al., 2001] Leroy, B., Medioni, G., Johnson, E., and Matthies, L. (2001). Crater detection for autonomous landing on asteroids. *Image and Vision Computing*, 19(11):787–792.

[Li and Lee, 1993] Li, C. H. and Lee, C. (1993). Minimum cross entropy thresholding. *Pattern recognition*, 26(4):617–625.

[Long et al., 2015] Long, J., Shelhamer, E., and Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440.

[Martins et al., 2009] Martins, R., Pina, P., Marques, J. S., and Silveira, M. (2009). Crater detection by a boosting approach. *IEEE Geoscience and Remote Sensing Letters*, 6(1):127–131.

[Melosh, 1988] Melosh, H. (1988). The rocky road to panspermia. *Nature*, 332(6166):687.

[Meng et al., 2009] Meng, D., Yunfeng, C., and Qingxian, W. (2009). Method of passive image based crater autonomous detection. *Chinese Journal of Aeronautics*, 22(3):301–306.

[Neukum, 1983] Neukum, G. (1983). Meteoritenbombardement und datierung planetarer oberflächen, habilitation dissertation for faculty membership. *Ludwig-Maximilians-University, Munich*, page 186.

[Neukum and Ivanov, 1994] Neukum, G. and Ivanov, B. (1994). Crater size distributions and impact probabilities on earth from lunar, terrestrial-planet, and asteroid cratering data. *Hazards due to Comets and Asteroids*, 1:359–416.

[Neukum et al., 1975] Neukum, G., König, B., Fechtig, H., and Storzer, D. (1975). Cratering in the earth-moon system-consequences for age determination by crater counting. In *Lunar and Planetary Science Conference Proceedings*, volume 6, pages 2597–2620.

[Opik, 1965] Opik, E. (1965). Mariner iv and craters on mars.

[Palafox et al., 2017] Palafox, L. F., Hamilton, C. W., Scheidt, S. P., and Alvarez, A. M. (2017). Automated detection of geological landforms on mars using convolutional neural networks. *Computers & geosciences*, 101:48–56.

[Polegubic, 2020] Polegubic, D. (2020). Landing site characterization for the pts lunar mission to the apollo 17 area. *Master thesis at the Technical University of Berlin.*

[Redmon et al.,] Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. You only look once: Unified, real-time object detection. pages 779–788. IEEE.

[Ren et al., 2015] Ren, S., He, K., Girshick, R., and Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99.

[Ridler et al., 1978] Ridler, T., Calvard, S., et al. (1978). Picture thresholding using an iterative selection method. *IEEE transactions on Systems, Man and Cybernetics*, 8(8):630–632.

[Robinson et al., 2010] Robinson, M., Brylow, S., Tschimmel, M., Humm, D., Lawrence, S., Thomas, P., Denevi, B., Bowman-Cisneros, E., Zerr, J., Ravine, M., et al. (2010). Lunar reconnaissance orbiter camera (lroc) instrument overview. *Space science reviews*, 150(1-4):81–124.

[Ronneberger et al., 2015] Ronneberger, O., Fischer, P., and Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer.

[Rumelhart et al., 1988] Rumelhart, D. E., Hinton, G. E., Williams, R. J., et al. (1988). Learning representations by back-propagating errors. *Cognitive modeling*, 5(3):1.

[Salamuniccar and Loncaric, 2010] Salamuniccar, G. and Loncaric, S. (2010). Method for crater detection from martian digital topography data using gradient value/orientation, morphometry, vote analysis, slip tuning, and calibration. *IEEE transactions on Geoscience and Remote Sensing*, 48(5):2317–2329.

[Sawabe et al., 2006] Sawabe, Y., Matsunaga, T., and Rokugawa, S. (2006). Automated detection and classification of lunar craters using multiple approaches. 37(1):21–27.

[Silburt et al., 2019] Silburt, A., Ali-Dib, M., Zhu, C., Jackson, A., Valencia, D., Kissin, Y., Tamayo, D., and Menou, K. (2019). Lunar crater identification via deep learning. *Icarus*, 317:27–38.

[Soderblom, 1970] Soderblom, L. A. (1970). *The distribution and ages of regional lithologies in the lunar maria.* PhD thesis, California Institute of Technology.

[Stepinski et al., 2012] Stepinski, T., Ding, W., and Vilalta, R. (2012). Detecting impact craters in planetary images using machine learning. In *Intelligent data analysis for real-life applications: theory and practice*, pages 146–159. IGI Global.

[Stepinski et al., 2009] Stepinski, T. F., Mendenhall, M. P., and Bue, B. D. (2009). Machine cataloging of impact craters on mars. *icarus*, 203(1):77–87.

[Stöffler and Ryder, 2001] Stöffler, D. and Ryder, G. (2001). Stratigraphy and isotope ages of lunar geologic units: Chronological standard for the inner solar system. In *Chronology and evolution of Mars*, pages 9–54. Springer.

[Szegedy et al.,] Szegedy, C., Wei Liu, Yangqing Jia, Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. Going deeper with convolutions. pages 1–9. IEEE.

[Uijlings et al., 2013] Uijlings, J. R., Van De Sande, K. E., Gevers, T., and Smeulders, A. W. (2013). Selective search for object recognition. *International journal of computer vision*, 104(2):154–171.

[Urbach and Stepinski, 2009] Urbach, E. R. and Stepinski, T. F. (2009). Automatic detection of sub-km craters in high resolution planetary images. *Planetary and Space Science*, 57(7):880–887.

[Vinogradova et al., 2002] Vinogradova, T., Burl, M., and Mjolsness, E. (2002). Training of a crater detection algorithm for mars crater imagery. In *Proceedings, IEEE Aerospace Conference*, volume 7, pages 7–7. IEEE.

[Viola et al., 2001] Viola, P., Jones, M., et al. (2001). Rapid object detection using a boosted cascade of simple features. *CVPR (1)*, 1(511-518):3.

[Wang et al., 2018] Wang, H., Jiang, J., and Zhang, G. (2018). Crateridnet: an end-to-end fully convolutional neural network for crater detection and identification in remotely sensed planetary images. *Remote Sensing*, 10(7):1067.

[Wetzler et al., 2005] Wetzler, P. G., Honda, R., Enke, B., Merline, W. J., Chapman, C. R., and Burl, M. C. (2005). Learning to detect small impact craters. In *2005 Seventh IEEE Workshops on Applications of Computer Vision (WACV/MOTION'05)-Volume 1*, volume 1, pages 178–184. IEEE.

[Williams et al., 2018] Williams, J.-P., van der Bogert, C. H., Pathare, A. V., Michael, G. G., Kirchoff, M. R., and Hiesinger, H. (2018). Dating very young planetary surfaces from crater statistics: A review of issues and challenges. *Meteoritics & Planetary Science*, 53(4):554–582.

[Yen et al., 1995] Yen, J.-C., Chang, F.-J., and Chang, S. (1995). A new criterion for automatic multilevel thresholding. *IEEE Transactions on Image Processing*, 4(3):370–378.