Technische Universität Berlin
Faculty:
Geodesy and Geoinformation Science

Professor:
Prof. Dr. Martin Kada

Master Thesis

Proposal
Lunar Crater Detection using Convolutional Neural Networks

Authors:
Tahir, Waqas

Spervisors:
Amgad Agoub

May 5, 2019

Contents

1 Introduction

Studying impact craters present a valuable information about the geology of planets and characteristics of the surface. It takes time and man power to detect craters from satellite images. A deep learning model will be applied to solve this problem which will reduce the cost of detection by eliminating skilled man power and saving time. In this model images will be subjected to convolutional operation by filters and such type of neural networks are known as Convolutional Neural Networks or CNNs. These are an architecture of supervised machine learning model 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. In this thesis a CNN algorithm will be customized, trained and tested for lunar crater detection. The proposed algorithm will be applied to images taken by Lunar Reconnaissance Orbiter irrespective of specific camera limitations. Images from all different angles and cameras are to be taken as dataset.

2 Background

The problem of detecting objects could be a complicated task specially when object background is complex. With advancements in computer technology it was no longer optimum to rely on manual detection of objects. With abundant data and limited time it was required to develop a mechanism that could solve such tasks in faster time compared to expensive manual operations and meanwhile limit man power. This gave rise to research in machine learning approaches to solve this problem.

Object detection and classification can be performed in supervised machine learning. In this technique an algorithm is trained by labeled images for desired classes of objects. The performance is then evaluated and tested using labeled images. The dataset is divided into 80% training and 20% test data. Labels are created by drawing a box around the object called a bounding box. The processes of supervised machine learning will be discussed in the later sections.

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:

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

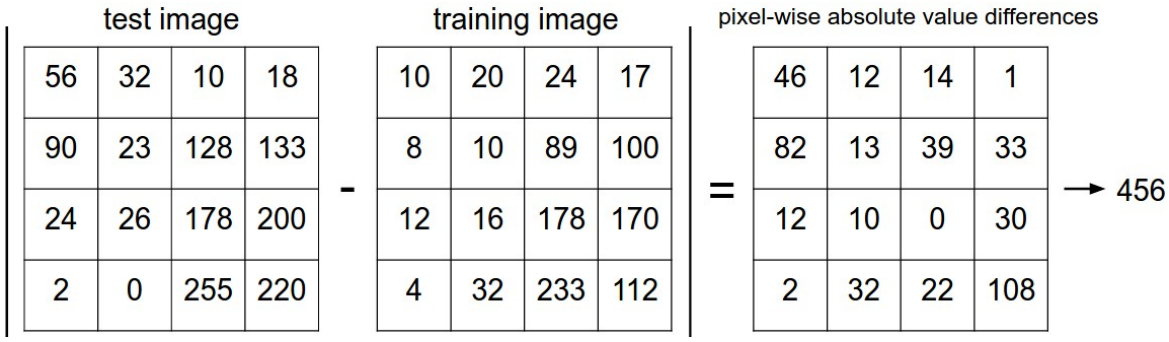

Figure 1: Example of two images which are represented as vectors and pixelwise difference is calculated to compare both images with L1 distance. This example shows one color channel. All the pixel-wise differences are added to denote a single digit value. If this value is close to zero then it indicates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

It is one of the simplest technique to classify an object. Eventually it will be extended to the entire Convolutional Neural Network. It has two major components. One is score function which does the mapping of raw data to the class scores and other one is a loss function that defines how different predicted score is from the ground truth labels. Typically loss function is higher initially and thus required to be minimized with respect to the score function parameters. This minimization is also referred to as optimization.

To define a score function which is the first component of linear classification, it is required to have information about number of images $\boldsymbol{N}$ with dimension $\boldsymbol{D}$ i.e. the size of image (pixels and channels) and distinct categories $\boldsymbol{K}$ which are the number of classes. A simplest possible function for linear mapping:

$$f(x_i, W, b) = Wx_i + b$$

Here x_i is the image which is flattened out to a single column vector of shape $[D \times 1]$. $\boldsymbol{W}$ is of size $[K \times D]$ and is often referred to as weights, whereas b is called bias vector with size $[K \times 1]$. Multiplication of matrices Wx_i gives separate classifiers in parallel where each classifier represents a row of weight matrix. The goal is to set the weights and bias parameters in a way that the output score match the ground truth labels of training data set as such that the correct classes has higher score than the incorrect classes.

2.3 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:

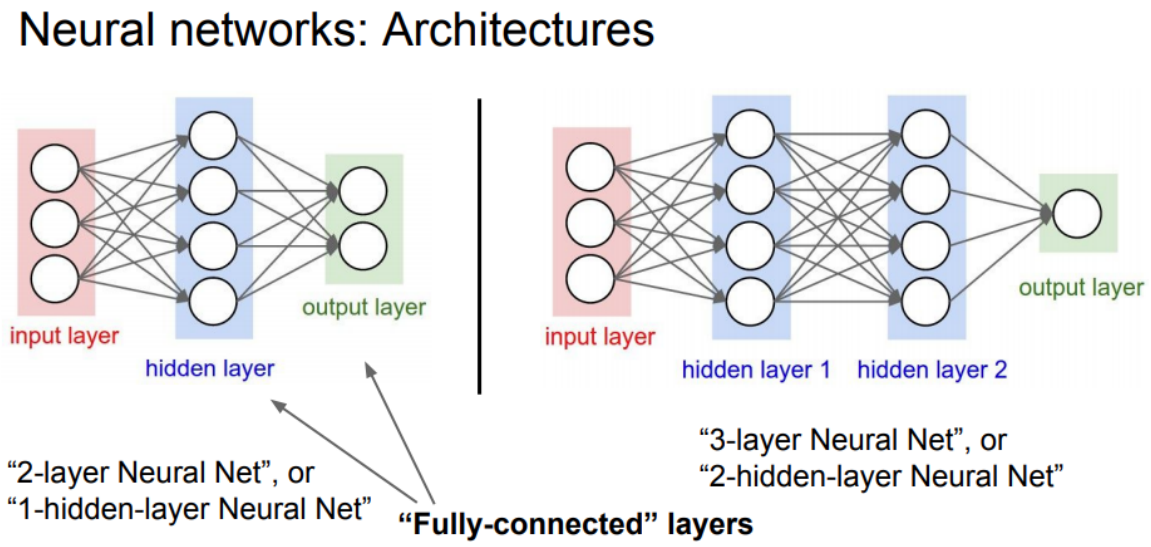

Figure 2: A typical Neural Network architecture

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.

2.4 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.

2.5 Concept of layers in CNNs

Following are the types of layers in a Convolutional Neural Network:

2.5.1 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.

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

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.

2.5.2 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)

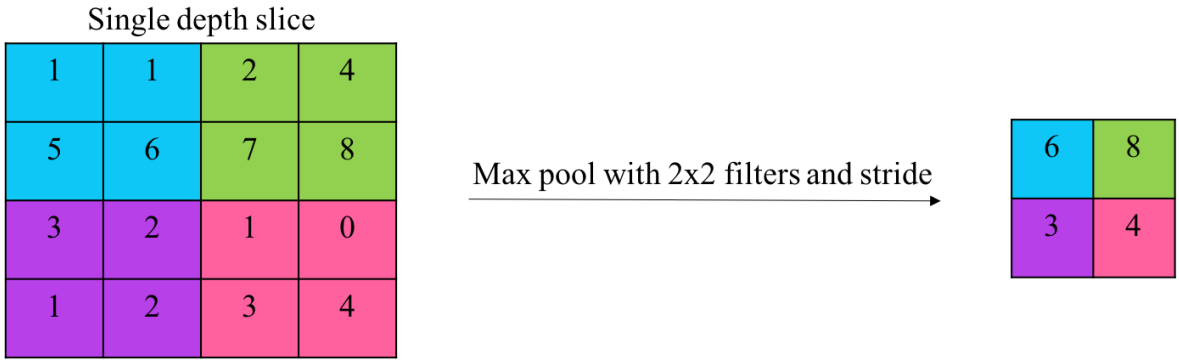

Figure 4: Pooling Operation

2.5.3 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:

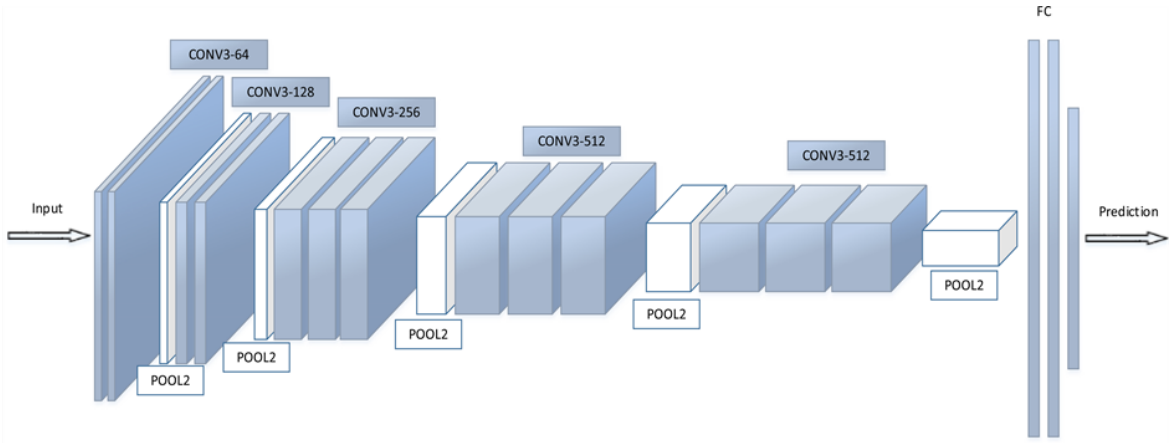

Figure 5: Convolutional Neural Network Architecture [El Khiyari and Wechsler, 2016]

3 Related Work

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 algorithm was applied on images acquired by Clementine and Apollo under different solar elevation. The accuracy was more than 80% compared to the manual detection results [Sawabe et al., 2006]. Accuracy rate was validated by comparing with crater count results of [Neukum et al., 1975]. 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.

The idea of machine learning existed well before early 2000s but because of lack of powerful GPU it was not possible to process the intense computation. In last decade, object detection techniques have been improved considerably. In 2013 a method was developed that refined the object detection performance from previously used approaches. The idea was to combine region proposals with CNN's and the method was referred to as R-CNN [Girshick et al.,]. Since then pixel level processing advancement in the algorithm has been made and in 2017 a team of Facebook researchers came up with a new algorithm that contributed to reduced processing time and increased accuracy (up to 99%) [He et al.,]. They came up with a method that does not only detect an object but also generates a high quality segmentation mask for each instance [He et al.,].

YOLO (You Look Only Once) is another method which was developed with an intention to detect objects at a greater speed than existing algorithms. YOLO lacks state of the art object detection accuracy achieved by other CNN algorithms i.e. Faster R-CNN and Mask R-CNN. But as it states you look only once, YOLO detects an object very quickly as compared to

Mask R-CNN but it struggles to localize some objects, specially small ones [Redmon et al.,]. This is because of its spatial constraints on bounding box predictions as every grid cell can have only one class.

4 Methodology

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*. Firstly, craters in images will be labeled. Every image will be subjected to image augmentation. This is done to achieve better training for the algorithm. Augmentation creates multiple images out of one image which means increased dataset and that let algorithm to have more training data and thus achieve better performance on test data. Most commonly used methods are mirroring and random cropping. Here in the dataset there are no RGB images hence color augmentation can not be performed.

Once this dataset is labeled and augmented then next step is to utilize this datset for training of the algorithm. In this process the selected amount of filters convolute over the image and outputs a layer with detected features. All the filters detect particular features and outputs a layer as shown in Figure 5. Number of layers generated are equal to amount of filters. These layers are then subjected to pooling function to downsize image and get the neurons of highest activation. A fully connected layer is the aggregation of all the layers followed by a loss function to determine if the image contains craters or not. Loss function determines the error in percentage and a process of back propagation adjusts the weights to minimize the error. After the detection is made with a minimal error possible, it would be interesting to come up with the evaluation method for challenging the performance of this algorithm.

References

[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.

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

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

[Krizhevsky et al., 2012] Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. 60(6):84–90.

[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.

[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.

[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.

[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.