**MAULANA AZAD**
**NATIONAL INSTITUTE OF TECHNOLOGY**
**BHOPAL INDIA, 462003**

---

---

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

# Image Thresholding

## Minor Project Report

### Semester 6

**Submitted by:**

| | |
|---|---|
| Vivek Kumar Ahirwar | 191112419 |
| Yuvraj Singh Rathore | 191112450 |
| Konjeti Sathwika | 191112432 |
| Ritik Parmar | 191112434 |

**Under the Guidance of**
Dr. Jyoti Bharti
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
**Session: 2022**

**MAULANA AZAD**

**NATIONAL INSTITUTE OF TECHNOLOGY**

**BHOPAL INDIA, 462003**

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

# CERTIFICATE

This is to certify that the project report carried out on "**Image Thresholding**" by the 3rd year students:

| | |
|---|---|
| Vivek Kumar Ahirwar | 191112419 |
| Yuvraj Singh Rathore | 191112450 |
| Konjeti Sathwika | 191112432 |
| Ritik Parmar | 191112434 |

Have successfully completed their project in partial fulfilment of their Degree in Bachelor of Technology in Computer Science and Engineering.

**Dr. Jyoti Bharti**
**(Minor Project Mentor)**

# DECLARATION

We, hereby declare that the following report which is being presented in the Minor Project Documentation Entitled as "**Image Thresholding**" is authentic documentation of our own original work and to the best of our knowledge. The following project and its report, in part or whole, has not been presented or submitted by us for any purpose in any other institute or organization. Any contribution made to the research by others, with whom we have worked at Maulana Azad National Institute of Technology, Bhopal or elsewhere, is explicitly acknowledged in the report.

| | |
|---|---|
| Vivek Kumar Ahirwar | 191112419 |
| Yuvraj Singh Rathore | 191112450 |
| Konjeti Sathwika | 191112432 |
| Ritik Parmar | 191112434 |

ACKNOWLEDGEMENT

With due respect, we express our deep sense of gratitude to our respected guide and coordinator Dr. Jyoti Bharti, for his valuable help and guidance. We are thankful for the encouragement that he has given us in completing this project successfully.

It is imperative for us to mention the fact that the report of the minor project could not have been accomplished without the periodic suggestions and advice of our project guide Dr. Jyoti Bharti and project coordinators Dr. Sanyam Shukla and Dr. Namita Tiwari.

We are grateful to our respected HOD, Dr. Nilay Khare and Dr. Meenu Chawla. We are also grateful to our respected director Dr. N. S. Raghuwanshi for permitting us to utilize all the necessary facilities of the college.

We are also thankful to all the other faculty, staff members and laboratory attendants of our department for their kind cooperation and help. Last but certainly not least; we would like to express our deep appreciation towards our family members and batch mates for providing the much-needed support and encouragement.

# ABSTRACT

Thresholding is the simplest method of image segmentation. From a grayscale image, thresholding can be used to create binary images. Thresholding methods are categorized into many groups based on the information the algorithm manipulates, in our project, we focus on different clustering-based Thresholding methods. We analyze various image pre-processing methods, categorize them, express their formulas and show a comparative study of all these methods.

As a part of preprocessing thresholding with segmentation is used for The description of the Otsu thresholding method along with histogram (based on intensity) and the result is documented. And according to our survey Otsu thresholding is better.

Image thresholding is used in many applications as a pre-processing step. For example, you may use it in medical image processing to reveal a tumour in a mammogram or localize a natural disaster in satellite images.

A problem with simple thresholding is that you have to manually specify the threshold value. We can manually check how good a threshold is by trying different values but it is tedious and it may break down in the real world.

To solve this problem, we are devising a method which can predict the perfect threshold for an image that can be useful for its real-life application. The Methods we have studied include all types of thresholding techniques such as Adaptive thresholding, Simple thresholding, Otsu's thresholding method, and Kapur’s thresholding method.

# Table of Contents

# Introduction

Thresholding is a type of image segmentation, where we change the pixels of an image to make the image easier to analyze. In thresholding, we convert an image from colour or grayscale into a binary image, i.e., one that is simply black and white. Most frequently, we use thresholding to select areas of interest in an image, while ignoring the parts we are not concerned with.

The downside of the simple thresholding technique is that we have to make an educated guess about the threshold *t* by inspecting the histogram. There are also automatic thresholding methods that can determine the threshold automatically for us. One such method is Otsu’s method. It is particularly useful for situations where the grayscale histogram of an image has two peaks that correspond to the background and objects of interest.

We have thoroughly discussed the major methods of image thresholding that are widely used to till now such as Adaptive thresholding, Simple Histogram based thresholding, Otsu's thresholding, Kapur’s thresholding etc.

Adaptive thresholding typically takes a grayscale or colour image as input and, in the simplest implementation, outputs a binary image representing the segmentation. For each pixel in the image, a threshold has to be calculated. If the pixel value is below the threshold it is set to the background value, otherwise, it assumes the foreground value.

The idea behind Ostu’s thresholding is that the method processes image histogram, segmenting the objects by minimization of the variance in each of the classes. Usually, this technique produces the appropriate results for bimodal images. The histogram of such an image contains two clearly expressed peaks, which represent different ranges of intensity values.

Similarly, Kapur’s method of thresholding is based on the entropy of an image. In this method, the criterion to select a suitable threshold is the maximization of Kapur's entropies based on the grey-level histogram. Kapur's original method is very time-consuming due to the inefficient formulation of the Kapur entropy and the exhaustive search in multilevel thresholding.

Our project mainly focuses on the use of a Deep learning model to predict the threshold of an image which can be used to get an accurate result as obtained by mathematical thresholding methods. The main Deep learning method that we have used is a Convolutional neural network ( CNN ).

A Convolutional Neural Network (ConvNet/CNN) is a Deep Learning algorithm that can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other. The pre-processing required in a ConvNet is much lower as compared to other classification algorithms. While in primitive methods filters are hand-engineered, with enough training, ConvNets can learn these filters/characteristics.

The model that we have used shows how thresholding can be used significantly to do segmentation and separate the image from its background. Image thresholding segmentation is a simple form of image segmentation. It is a way to create a binary or multi-colour image based on setting a threshold value on the pixel intensity of the original image. Recently, due to the success of deep learning models in a wide range of vision applications, there has been a substantial amount of work aimed at developing image segmentation approaches using deep learning models. We investigate the similarity, strengths and challenges of these deep learning models, examine the most widely used datasets, report performances, and discuss promising future research directions in this area.

# Literature review and survey

## 1. Thresholding

Thresholding is the process of creating a black and white image out of a greyscale image by setting exactly those pixels to white whose value is above a given threshold, setting other pixels to black. Grayscale images by thresholding can be used to create binary images.

The simple thresholding method involves replacing each pixel in an image with a black pixel if the image intensity src(x,y) is less than some fixed constant T (that is, src(x,y) <T), or a white pixel if the image intensity is greater than that constant. The input to such a thresholding algorithm is usually a grayscale image and a threshold. The output is a binary image.

Image thresholding is used in many applications as a pre-processing step. For example, you may use it in medical image processing to reveal a tumour in a mammogram or localize a natural disaster in satellite images.

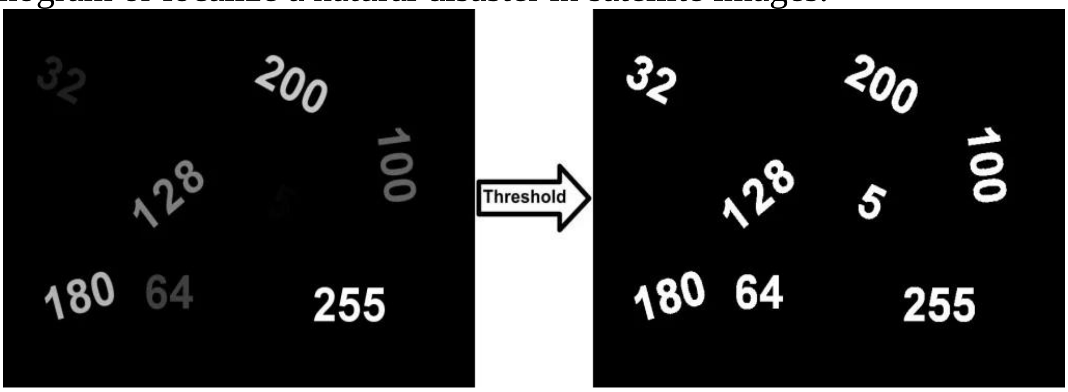



*Figure 1: An Example of Binarization of Image*

Image thresholding is sub-divided into the local and global image thresholding algorithms.

### i. Global Thresholding

Global thresholding algorithms take a source image (src) and a threshold value (thresh) as input, and produce an output image (dst), by comparing the pixel intensity at the source pixel location (x,y) to the threshold. If src(x,y) > thresh, then dst(x,y) is assigned some value. Otherwise, dst(x,y) is assigned some other value.
Global thresholding is based on the assumption that the image has a bimodal histogram and, therefore, the object can be extracted from the background by a simple operation that compares image values with a threshold value T.
The object and background pixels have grey levels grouped into two dominant modes. One obvious way to extract the object from the background is to select a threshold T that separates these modes.

The thresholded image g(x,y) is defined as g(x, y)

$$g(x,y) = \begin{cases} 1 & \text{if } (x,y) > T \\ 0 & \text{if } (x,y) \leq T \end{cases}.$$

The result of thresholding is a binary image, where pixels with an intensity value of 1 correspond to objects, whereas pixels with a value of 0 correspond to the background.

ii. **Local Thresholding**

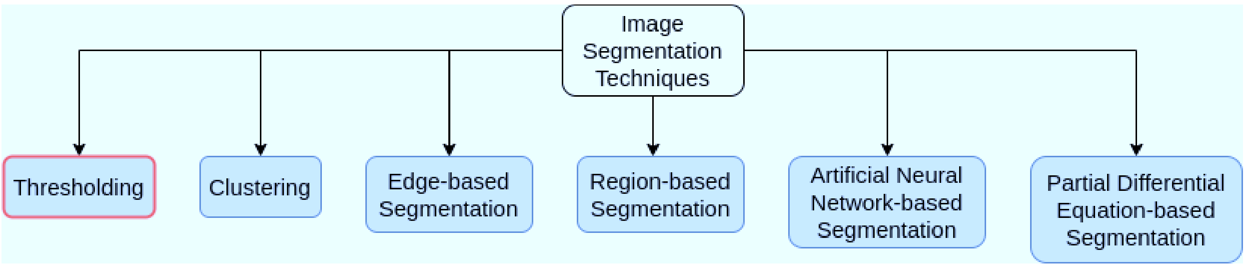
In local thresholding, some characteristics of some local image areas (e.g., the local contrast) may be used to choose a different threshold for different parts of the image. Local adaptive thresholding is used to convert an image consisting of greyscale pixels to just black and white scale pixels. Usually, a pixel value of 0 represents white and the value 255 represents black with the numbers from 1 to 254 representing different grey levels. Unlike the global thresholding technique, local adaptive thresholding chooses different threshold values for every pixel in the image based on an analysis of its neighbouring pixels. This is to allow images with varying contrast levels where a global thresholding technique will not work satisfactorily. There are several different forms of adaptive thresholding algorithms reported in the image processing literature.

## 2. Comparing Image Thresholding and Segmentation

Segmentation is dividing an image into different segments or groups of pixels. It helps us in many ways such as object detection, video surveillance, traffic navigation etc.

In that sense, image thresholding is the simplest kind of image segmentation because it partitions the image into two groups of pixels — white for the foreground, and black for the background.

The figure below shows different types of segmentation algorithms:



# 3. Different Thresholding Techniques

## (i) Single-Thresholding (Histogram Based)

It is the simplest approach to image segmentation. Here we analyse peaks and valleys of a smoothened histogram to find the threshold of an image. Suppose, if we have images where every pixel has been coded with eight bits, then we have intensity varying from 0 to 255. We can decide the threshold following some criteria. For example, we have threshold level 128 so that it decides that all the pixels having intensity values greater than 128 belong to some regions and that intensity values less than 128, belong to some other region. Let an image be f(x, y). Suppose that this image consists of a dark object against a bright background or vice-versa. Therefore, intensity concentrates mainly on two regions, one towards the darker side (or lower intensity) and the other towards the brighter side (or higher intensity).

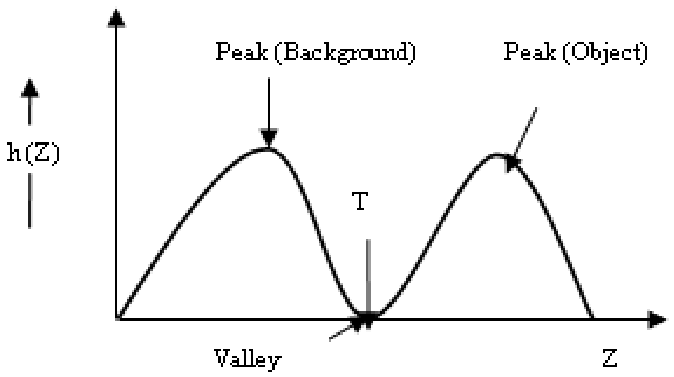



*Figure 2: Bimodal Image Histogram*

The figure given above shows a histogram of a bimodal image. Here T is the threshold of the image with a peak of the background and object lying on either side of the threshold.

## (ii) Adaptive Thresholding

In this method, we subdivide the image into several smaller sub-images. Assume that in each of these sub-images, the intensity will be more or less uniform. We can easily find out the threshold value for each of these sub-images. Using this threshold value, we can threshold the sub-images and take the union of all these threshold values, it gives the final threshold value. In this case, the threshold selection is position-dependent. That is why it is called adaptive thresholding.

In OpenCV, you can perform an Adaptive threshold operation on an image using the method **adaptive threshold()** of the **Imgproc** class.

Depending on your project, leveraging adaptive thresholding can enable you to:

(i) Obtain better segmentation than using global thresholding methods, such as basic thresholding and Otsu thresholding
(iii) Avoid the time consuming and computationally expensive process of training a dedicated Mask R-CNN or U-Net segmentation network

### (iv) Otsu's Thresholding

The method processes image histogram, segmenting the objects by minimization of the variance in each of the classes. Usually, this technique produces the appropriate results for bimodal images. The histogram of such an image contains two clearly expressed peaks, which represent different ranges of intensity values.

This method assumes histogram (and the image) are bimodal and uniform illumination (implicitly), so the bimodal brightness behaviour arises from object appearance differences only.

The core idea is to separate the image histogram into two clusters with a threshold defined as a result of minimization of the weighted variance of these classes denoted by $\sigma_w^2(t)$.

The whole computation equation can be described as: $\sigma_w^2(t) = w_1(t)\sigma_1^2(t) + w_2(t)\sigma_2^2(t)$, where $w_1(t), w_2(t)$ are the probabilities of the two classes divided by a threshold $t$, which value is within the range from 0 to 255 inclusively.

There are two options to find the threshold. The first is to minimize the within-class variance $\sigma_w^2(t)$ defined above, the second is to maximize the between-class variance using the expression below: $\sigma_b^2(t) = w_1(t)w_2(t)[\mu_1(t) - \mu_2(t)]^2$, where $\mu_i$ is a mean of class $i$.

The probability $P$ is calculated for each pixel value in two separated clusters $C_1, C_2$ using the cluster probability functions expressed as:

$w_1(t) = \sum_{i=1}^{t} P(i)$,
$w_2(t) = \sum_{i=t+1}^{I} P(i)$

It should be noted that the image can present as intensity function $f(x, y)$, which values are grey-level. The quantity of the pixels with a specified grey-level $i$ denotes by $i$. The general number of pixels in the image is $n$.

Thus, the probability of grey-level $i$ occurrence is: $P(i) = \frac{n_i}{n}$.

The pixel intensity values for the $C_1$ are in $[1, t]$ and for $C_2$ are in $[t + 1, I]$, where $I$ is the maximum pixel value (255).
The next phase is to obtain the means for $C_1, C_2$, which are denoted by $\mu_1(t), \mu_2(t)$ appropriately:
$\mu_1(t) = \sum_{i=1}^{t} \frac{iP(i)}{w_1(t)}$,

$\mu_2(t) = \sum_{i=t+1}^{I} \frac{iP(i)}{w_2(t)}$

Now let's remember the above equation of the within-classes weighted variance. We will find the rest of its components ($\sigma_1^2, \sigma_2^2$) by mixing all the obtained above ingredients:

$\sigma_1^2(t) = \sum_{i=1}^{t} [i - \mu_1(t)]^2 \frac{P(i)}{w_1(t)}$,

$\sigma_2^2(t) = \sum_{i=t+1}^{I} [i - \mu_2(t)]^2 \frac{P(i)}{w_2(t)}$.

It should be noted that if the threshold was chosen incorrectly the variance of some classes would be large. To get the total variance we simply need to summarize the within-class and between-class variances: $\sigma_T^2 = \sigma_w^2(t) + \sigma_b^2(t)$, where $\sigma_b^2(t) = w_1(t)w_2(t)[\mu_1(t) - \mu_2(t)]^2$. The total variance of the image ($\sigma_T^2$) does not depend on the threshold.

Thus, the general algorithm's flow for the between-class variance maximization option can be represented in the following way:

1. calculate the histogram and intensity level probabilities
2. initialize $w_i(0), \mu_i(0)$
3. iterate over possible thresholds: $t = 0, ..., max\_intensity$
   - update the values of $w_i, \mu_i$, where $w_i$ is a probability and $\mu_i$ is a mean of class $i$
   - calculate the between-class variance value $\sigma_b^2(t)$
4. the final threshold is the maximum $\sigma_b^2(t)$ value

### (v) Kapur's Method

Kapur has developed a method for bimodal or bi-level thresholding which can be described as: Let L be the grey level of the image under consideration and the grey level ranges between {0,1,2… (L−1)}. Then, the prevalence probability of grey level z is given by:

Pz = k(z)N; For (0≤z≤(L−1)).

Where k(z) signifies the number of pixels corresponding to each grey level, L and the total number of pixels is represented by N which is denoted as:

N=ΣL−z=0k(z)

The objective is to maximize the fitness function as follows:

f(s)=H0+H1

where,

$$H_0 = -\Sigma_{z=0}^{s-1} \frac{P_z}{w_0} \ln \frac{P_z}{w_0}, \qquad w_0 = \Sigma_{z=0}^{s-1} P_z \text{ and}$$
$$H_1 = -\Sigma_{z=s}^{s-1} \frac{P_z}{w_1} \ln \frac{P_z}{w_1}, \qquad w_0 = \Sigma_{z=s}^{s-1} P_z$$

The above method can be used to evaluate multi-level threshold valley points and is described as an N-dimensional optimization problem, for choosing r threshold values for a given image [s1,s2,…., sn]. Here also our objective is to maximize the function:

$$\mathrm{f}(s_1, s_2, s_3, \ldots, s_n]) = H_0 + H_1 + H_2 + \cdots + H_n$$

Where,

$$H_0 = -\Sigma_{z=0}^{s_1-1} \frac{P_z}{w_0} \ln \frac{P_z}{w_0}, \qquad w_0 = \Sigma_{z=0}^{s-1} P_z$$
$$H_1 = -\Sigma_{z=s1}^{s_2-1} \frac{P_z}{w_0} \ln \frac{P_z}{w_0}, \qquad w_0 = \Sigma_{z=s1}^{s_2-1} P_z$$
$$H_n = -\Sigma_{z=sn}^{L-1} \frac{P_z}{w_0} \ln \frac{P_z}{w_0}, \qquad w_0 = \Sigma_{z=sn}^{L-1} P_z$$

# Gaps Identified

1. **Otsu's method works properly on Bimodular images**

If two dominant modes characterize the image histogram, it is called a bimodal histogram. Only one threshold is enough for partitioning the image.
If for example an image is composed of two types of dark objects on a light background, three or more dominant modes characterize the image histogram.
Single-thresholding uses the bi-modal histogram which gives the single threshold value. It is a very simple approach and it works well in the case of uniform images. This technique has failed when there is a multi-modal histogram. Otsu's and other mathematical thresholding algorithms work properly on finding a single threshold of an image. But when we take Multimodular images they don't work properly because in these cases we need multiple thresholds for an image to be segmented properly.

2. **These methods don't tell about the accuracy of a threshold**

If we sub-divide the image and take the separate threshold for each region and then, take the union of all these thresholds, we will get an adaptive threshold. But, it also does not tell the accuracy of the threshold. So, to find the accurate threshold value, we use the optimal thresholding. In that case, we find the optimum threshold. Suppose, when there are very small object pixels and a large number of background pixels in an image, then, optimal thresholding does not work well because it is difficult to obtain the boundary itself. To overcome this problem, we use the local properties (Laplacian and Gradient) of the pixels to find the threshold value which is called local thresholding. In short, all these techniques work well in some situations and do not work well in other situations. Therefore, based on the given image, we can segment the image by using these thresholding techniques. So instead of these mathematically based methods to find the threshold, we can use CNN models (eg MASK-R CNN) to compare the accuracy of these methods and provide a good mask to separate the background and foreground of the images.

3. **Different intensities of an image affect threshold of an image overlay**

Thresholding means converting an image into a binary format. It is important for image processing. Because sometimes people need separation of the dark and light regions. Thresholding images can separate the dark and light sides of the colourful image. Another reason is, that you can convert binary images in any format using that. To make the image binary, you have to go to the adjustment tool and select "Threshold". That's an easy way to do this.

# Proposed work and methodology

## 1. Proposed work

The goal of this project is to find the optimal image threshold using a technique inspired by deep learning models to binarize the greyscale image. The image thresholding technique has various applications in real life as well as in scientific research. Some of the applications are object detection, segmentation, background extraction and removal and various medial applications, for example, finding retinal veins from the image of the retina or detecting tumors etc.

## 2. Methodology

The objective of our project is to find a threshold for a greyscale image to binarize it using a convolution neural network (CNN).
So, first, how does CNN work:
Convolutional Neural Networks (CNN or ConvNet) are complex feed-forward neural networks. CNNs are used for image classification and recognition because of their high accuracy. The CNN follows a hierarchical model which works on building a network, like a funnel, and finally gives out a fully-connected layer where all the neurons are connected and the output is processed. CNN describes the concept of hierarchical feature detectors in a biologically inspired manner.

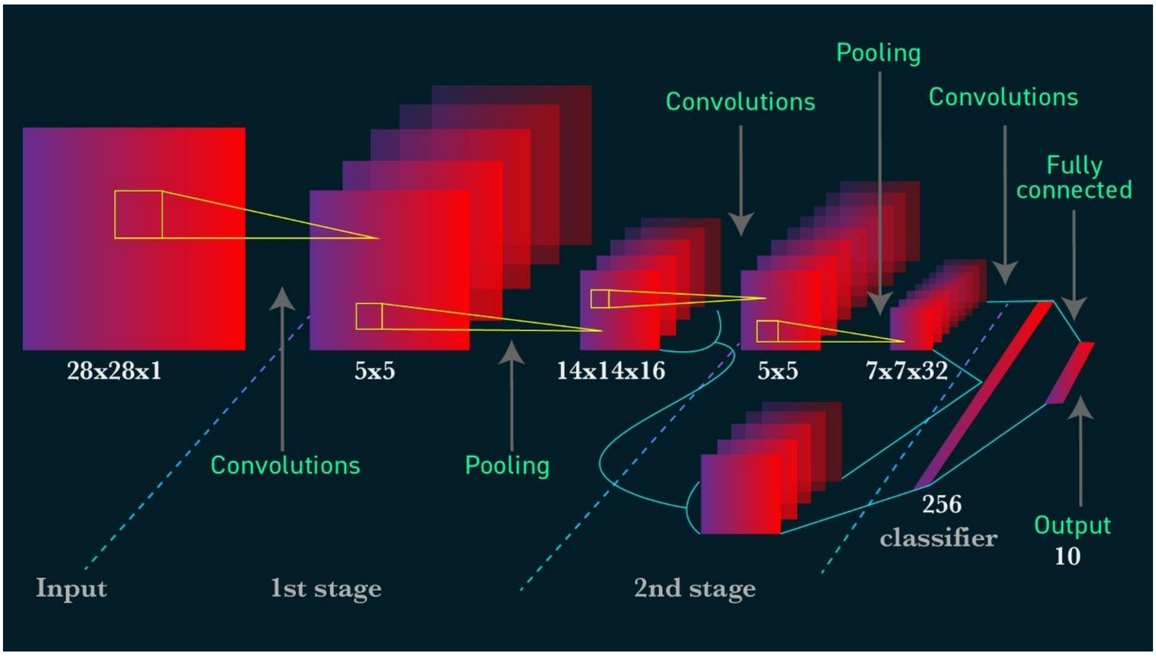



*Figure 3: General CNN Architecture*

**Convolution Layer:** A convolutional layer is the main building block of a CNN. It contains a set of filters (or kernels), parameters of which are to be learned throughout the training. The size of the filters is usually smaller than the actual image. Each filter convolves with the image and creates an activation map.

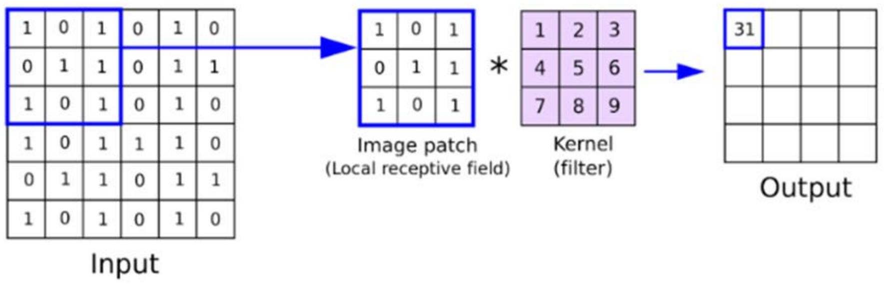



*Figure 4: : Convolutional Layer showing Convolution Operation*

**ReLU Layer:** The rectified linear activation function or ReLU for short is a piecewise linear function that will output the input directly if it is positive, otherwise, it will output zero. So, after passing through the ReLU layer only positive inputs remain.

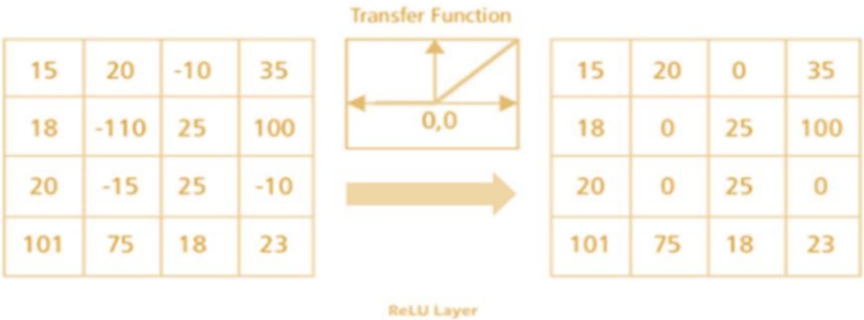



*Figure 5: ReLU Layer performing ReLU operation*

**Pooling Layers:** Pooling layers are performed after applying convolution and ReLu operations to reduce the dimensions of the feature maps. Thus, it reduces the number of parameters to learn and the amount of computation performed in the network. The pooling layer summarizes the features present in a region of the feature map generated by a convolution layer. Pooling has two types: Max Pooling and Average Pooling.

**Max Pooling** slides a window over the input image and stores the max value of the window in the output.

**Average Pooling:** slides a window over the input and stores the average value of the window in the output.

It helps to reduce computation by discarding 75% of neurons (assuming 2x2 filters with the stride of 2) and it makes feature detectors more robust.

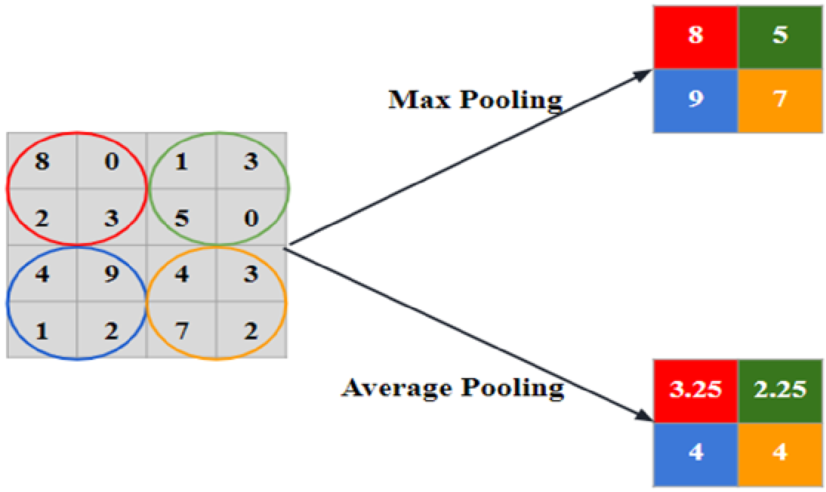



*Figure 6: Pooling layer showing (a)Max Pooling and (b) Average Pooling*

**Stochastic Pooling:** In this pooling layer ,it integrates the advantages of max-pooling, average-pooling and stochastic pooling is introduced. The proposed pooling is designed to balance the advantages and disadvantages of max-pooling and average-pooling by using the degree of sparsity of activations and a control function to obtain an optimized representative feature value ranging from average value to maximum value of a pooling region. The optimized representative feature value is employed for probability weights assignment of activations in normal distribution. The proposed pooling also adopts weighted random sampling with a reservoir for the sampling process to preserve the advantages of stochastic pooling. This proposed pooling is evaluated on several standard datasets in deep learning framework to compare with various classic pooling methods. Experimental results show that it has good performance on improving recognition accuracy. The influence of changes to the feature parameter on recognition accuracy is also investigated.

## 3. Procedure

The concept of pooling is used to find the threshold for an image. In this procedure, repetitive max-pooling layer and average pooling are applied with a fixed value of stride until a matrix of size 1x1 is obtained. the value in the 1x1 matrix is our desired value threshold value. On comparing the results, the max-pooling was not giving fair results.

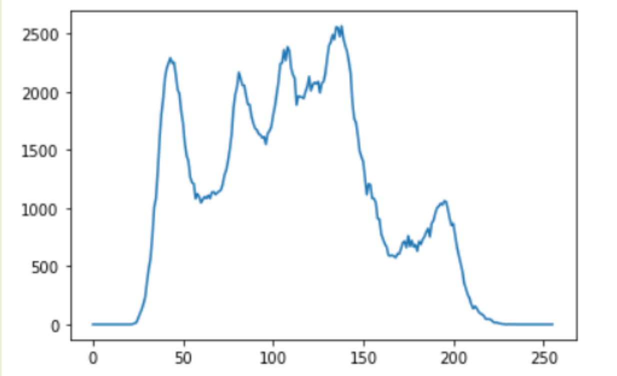
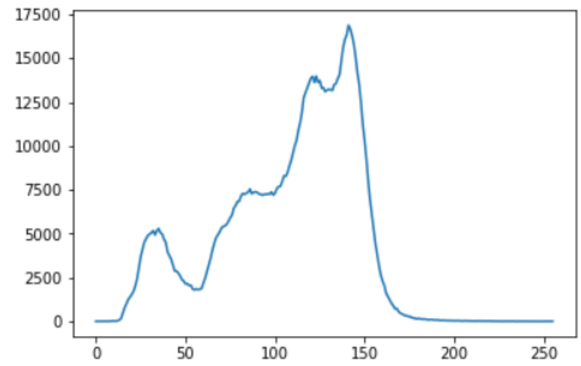
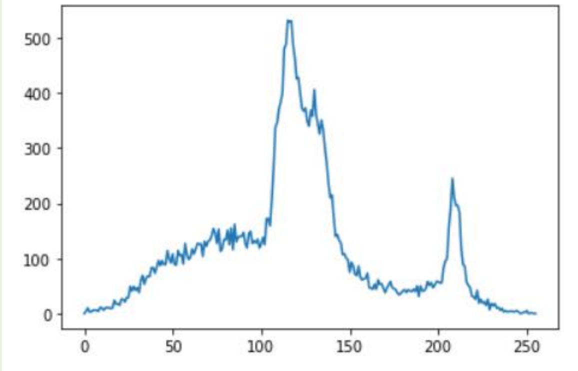
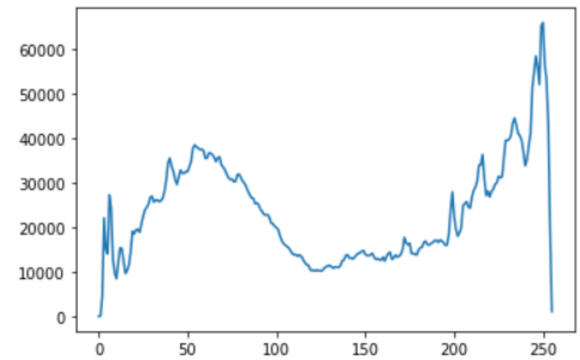
Now to further optimize the results of average pooling, the last-second matrix of last-second iteration in repetitive pooling operation was used. The statistical operation was applied to the values of the matrix. Mean, median and mode values of elements of the matrix were calculated. The threshold values obtained from these are used to binarize the image. Results were compared to find the best threshold value.

## 4. Libraries Used:

- Numpy
- Pandas
- Matplotlib
- cv2
- Keras, Tensorflow
- Scipy

# Result and Discussion

## Histograms of input images:

<table>
<tr><th>Input Image</th><th>Histogram</th></tr>
<tr><td></td><td>
</td></tr>
<tr><td></td><td>
</td></tr>
<tr><td></td><td>
</td></tr>
<tr><td></td><td>
</td></tr>
</table>

**Threshold values based on the Histogram analysis method:**

| Input Image | Otsu's Thresholding | Threshold based on Histogram analysis |
|---|---|---|
|  | <br>Otsu’s Threshold = 109 | <br>Threshold value: 59 |
|  | <br>Otsu’s Threshold = 96 | <br>Threshold value: 132 |
|  | <br>Otsu’s Threshold = 144 | <br>Threshold value: 103 |
|  | <br>Otsu’s Threshold = 138 | <br>Threshold value: 21 |

In **Max Pooling** most of the outcomes were returning the maximum possible value of the intensity of grey i.e., 255. So Max pooling was giving meaningful results for Image Thresholding.

**Average Pooling Outcomes:**

| Input Image | Otsu's Thresholding | Proposed method: Average Pooling(Optimal) |
|---|---|---|
|  |  Otsu’s Threshold = 109 |  Threshold value: 110 |
|  |  Otsu’s Threshold = 96 |  Threshold value: 100 |
|  |  Otsu’s Threshold = 144 |  Threshold value: 116 |
|  |  Otsu’s Threshold = 138 |  Threshold value: 116 |

**Statistical operation on Average Pooling Outcomes:**

| Input Image | Otsu's Thresholding | Proposed method: Average Pooling (Mean) | Proposed method: Average Pooling (Median) | Proposed method: Average Pooling (Mode) |
|---|---|---|---|---|
|  |  Otsu’s Threshold = 109 |  Mean: 110.204 |  Median: 115.073 |  Mode: 87.014 |
|  |  Otsu’s Threshold = 96 |  Mean: 100.189 |  Median: 100.189 |  Mode: 97.658 |
|  |  Otsu’s Threshold = 144 |  Mean: 116.174 |  Median: 122.768 |  Mode: 77.230 |
|  |  Otsu’s Threshold = 138 |  Mean: 116.006 |  Median: 116.006 |  Mode: 95.813 |

**Stochastic Pooling**

| Input Image | Otsu's Thresholding | Stochastic Pooling |
|---|---|---|
|  |  Otsu’s Threshold = 109 |  Threshold value: 129 |
|  |  Otsu’s Threshold = 96 |  Threshold value: 125 |
|  |  Otsu’s Threshold = 144 |  Threshold value: 142 |
|  |  Otsu’s Threshold = 138 |  Threshold value: 145 |

# Conclusion

We have presented the comparative study of various available Image Thresholding techniques such as Single Thresholding (histogram-based), Adaptive Thresholding, Otsu's Method, and Kapur's Method, from all these methods, Otsu's method performs best when the histogram has a bimodal distribution with a deep and sharp valley between the two peaks. Thresholding is a type of image segmentation technique. As far as the current survey we conducted there is no available machine learning or deep learning-based method to find the threshold for an image to binarize it. We have proposed a method based on the Pooling layer of CNN architecture to find image thresholds. The outcomes from our proposed method: average pooling with the statistical operation was quite good and, in some cases, it was performing better than the best available method i.e., Otsu's Method. Image thresholding has applications in various fields like satellite imaging, medical purposes, object detection background removal etc.

# References

[1] P. K. Sahoo, S. Soltani and A. K. C. Wong, "A Survey of Thresholding Techniques.," Computer Vision, Graphics, and Image Processing 41. 233-260. 10.1016/0734-189X(88)90022-9, 1988.

[2] P. Guruprasad, "OVERVIEW OF DIFFERENT THRESHOLDING METHODS IN IMAGE PROCESSING," in *TEQIP Sponsored 3rd National Conference on ETACC*, 2020.

[3] G. Kaur and K. Singh, "A Comparative Study of Image Segmentation using Thresholding Techniques," in *National Conference on Advanced Computing Technologies (NCACT-2013)*, 2013.

[4] R. Singh, P. Agarwal, M. Kashyap and M. Bhattacharya, "Kapur's and Otsu's based optimal multilevel image thresholding using social spider and firefly algorithm," in *International Conference on Communication and Signal Processing (ICCSP)*, 2016.

[5] N. Otsu, "A threshold selection method from gray level histograms," in *IEEE Transactions on Systems, Man, and Cybernetics 9*, 1979.

[6] X. Du, Y. Cai, S. Wang and L. Zhang, "Overview of deep learning," in *31st Youth Academic Annual Conference of Chinese Association of Automation (YAC)*, 2016.

[7] S. Albawi, T. A. Mohammed and S. Al-Zawi, "Understanding of a convolutional neural network," in *International Conference on Engineering and Technology (ICET)*, 2017.

[8] J. Rogowska, Handbook of Medical Image Processing and Analysis (Second Edition), 2009.

[9] H. Kiani, R. Safabakhsh and E. Khadangi, "Fast recursive segmentation algorithm based on Kapur's entropy," in *2nd International Conference on Computer, Control and Communication*, 2009.

[10] Minaee, Shervin, Boykov, Yuri, Porikli, Fatih, Plaza, Antonio, Kehtarnavaz, Nasser, Terzopoulos and Demetri, "Image Segmentation Using Deep Learning: A Survey," in *arXiv*, 2020.

[11] W. Zhao, Z. ye, M. Wang, L. Ma and W. Liu, "An image threholding approach based on cuckoo search algorithm and 2D maximum entropy," in *2015 IEEE 8th International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)*, 2015.

[12] L. Caponetti and G. Castellano, Fuzzy Logic for Image Processing, 2017.

[13] M. Bidlo, "Evolution of Approximate Functions for Image Thresholding," in *2021 IEEE Symposium Series on Computational Intelligence (SSCI)*, 2021.

[14] A. Prahara, A. Pranolo, N. Anwar and Y. Mao, "Parallel Approach of Adaptive Image Thresholding Algorithm on GPU," *Knowledge Engineering and Data Science*, 2022.

[15] T. Akan--R.Farshi and R. Demirci, "Multilevel image thresholding with multimodal optimization," *Multimedia Tools and Applications*, 2021.

[16] L. Li, X. Huang, S. Qian, Z. Li, S. Li and R. F. Mansour, "Fuzzy Hybrid Coyote Optimization Algorithm for Image Thresholding," *Computers, Materials and Continua* , 2022.

[17] P. K. N.S.R. and P. R. P.V.G.D., "Evolutionary Image Thresholding for Image Segmentation," in *International Journal of Computer Vision and Image Processing*, 2019.

[18] S. Mahajan, N. Mittal and A. Pandit, "Image segmentation using multilevel thresholding based on type II fuzzy entropy and marine predators algorithm," *Multimedia Tools and Applications* , 2021.

[19] M. Yufang and Y. Zhang, "FADIT: Fast Document Image Thresholding," *Algorithms*, 2020.

[20] N. Razmjooy and R. S. Majdar, "A New Image Threshold Technique based on Metaheuristics," in *Conference: 5th National Conference on Soft Computing in Electrical and Computer Engineering (SCECE2020)*, Ardabil, Iran, 2021.