**Malaria Dataset :**

**Dataset** from : <https://www.kaggle.com/iarunava/cell-images-for-detecting-malaria>

**Image Preprocessing Techniques** from :

<https://www.kaggle.com/sharansh12345/99-7-accuracy-for-malaria-dataset>

This link also had a Code, but It was not correct. So, I used Different CNN Model and only taken Image Preprocessing Techniques from here.

We got **accuracy of 95%** on Malaria Detection ( Parasitize or Unparasitized)

I tried **Two CNN Models** : **BUT I USED 2ND ONE in this github python file.**

1. **Simple Model:**

In which 5 CNN Blocks were used.

Used Activation function of RELU in all layers except in the last layer which used sigmoid.

We have used Pooling Operation Alternatively on each CNN block.

We used dropout layer for regularization purposes and used it at the end of each CNN Layer.

After each CNN Block our Image size reduced as we did not Pad, and we used stride of 2.

Glorot Uniform Weight initializer was used.

So this CNN Architecture acted as an Encoder and we also used More channels after each CNN (16 , 32, 64 ,128 , 256 )

1. **Complex Architecture :**

Same CNN Architecture as used in the WBC Classification Task.

**Image Preprocessing:**

Used a Kernel , as shown below , which has resulted in making the Parasitized Portion of the Malaria Cells MORE VISIBLE/STAND OUT thus CNN Model had learnt the features and weights quickly( in 4 epochs ) and we got 95% accuracy.

**[0 , -1 , 0 ]**

**[-1 , 6 , -1]**

**[0 , -1 , 0]**

**Step1:**

**Applied Kernel**

**Convert BGR to YUV**

**Used equalizeHist on YUV Image data ( on channel 0 and rest of channels remain same)**

**Then Converted it that YUV to RGB**

**A picture containing shape

Description automatically generated**

**Histogram Equilizer:**

Consider an image whose pixel values are confined to some specific range of values only. For eg, brighter image will have all pixels confined to high values. But a good image will have pixels from all regions of the image. So you need to stretch this histogram to either ends (as given in below image, from **wikipedia**) and that is what Histogram Equalization does (in simple words). This normally **improves the contrast of the image.**

**WE APPLY HISTOGRAM\_EQUILIZER on this 0th CHANNEL on YUV Image dataset**

**ONLY ON ONE CHANNEL , WE APPLY HISTOGRAM\_EQUILIZER**

**Chart, histogram

Description automatically generated**

**This is 0th Channel of YUV Image (Taken a image randomly and to see how 0th channel of YUV Looks)**

**Original Images:**

Background pattern

Description automatically generated with medium confidence

**YUV Images:**

**Shape, square

Description automatically generated**

**After Image Preprocessing , we got:**

A picture containing text

Description automatically generated