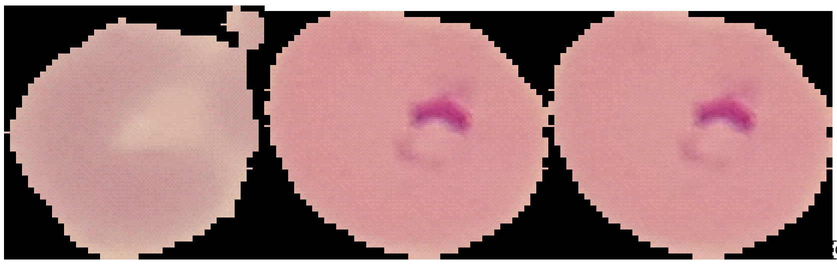
Deep learning is an AI function that can work as human brain and can make decisions from processed data without human supervision.Convolutional networks are simply neural networks that use convolution in place of general matrix multiplication in at least one of their layers. Convolutional layer, pooling layer, dense layer and fully-connected layer are some of the layers of CNN. Deep learning called convolutional neural network was used by detecting and deploying Image Cells that contain Malaria or not.

Malaria is a life-threatening disease caused by parasites that are transmitted to humans by bites of infected Anopheles female mosquitoes. It can be prevented, and cured. There are 212 Million cases of malaria and 435000 deaths according to WHO. Early diagnosis and malaria care can help avoid deaths. The seriousness of malaria is prevalent worldwide, particularly in tropical regions.



.Dataset: https://www.kaggle.com/iarunava/cell-images-for-detecting-malaria

The dataset of labeled and preprocessed images to train and evaluate our model. The dataset has a malaria dataset of 27,558 cell images with an equal number of parasitized and uninfected cells. A level-set based algorithm was applied to detect and segment the red blood cells. The images were collected and annotated by medical professionals;

We split the dataset into 2 parts. The test and train manually. The train dataset has `13513 images and test dataset has 14027 images.`

Methodology:

Convolutional neural network: CNN is a hierarchical feed-forward network which consists of different types of layers such as convolutional layers, pooling layers, dense layers, dropout, flatten and so on. Different types of activation functions such as relu, sigmoid, softamax, tanh can be used with CNN. In our model, we have used convolutional layers, pooling layers, dense layer, dropout and relu and sigmoid was used as activation functions.

Convolutional layers: The layer consists with number of filters. If the input size is n×n and f×f kernel is used, then the output is the following equation.

$$(n \times \mathrm{n}) * (\mathrm{f} \times \mathrm{f}) = (\mathrm{n} - \mathrm{f} + 1) * (\mathrm{n} - \mathrm{f} + 1)$$

Pooling layer: Pooling layer is also known as down sampling layer. There are two kinds of pooling layer, 1) max pooling and 2) average pooling. In this paper, max pooling was used with same padding.

To do same padding the equation is:

$$n + 2p - f + 1 = n$$
$$p = \frac{f - 1}{2}$$

Here, n is the input, p is padding size, f is kernel size.

After applying pooling layer with same padding the output should be: $(n + 2p - f)$+1

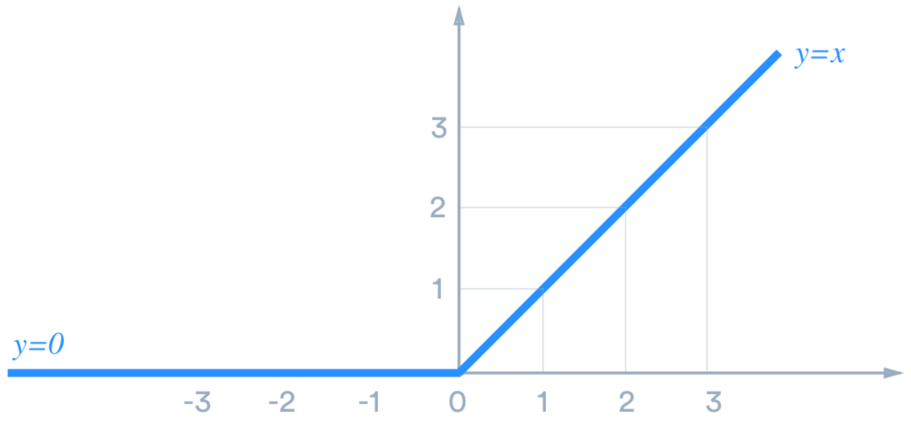
Dropout: Dropout layer is used to reduce overfitting. If dropout is added, it will randomly ignore some subset of nodes in a given layer during training.

Flatten layer: Flatten layer is in between convolutional layers and fully connected layer which transforms a two-dimensional matrix of features into a vector that can be fed into a fully connected neural network classifier.

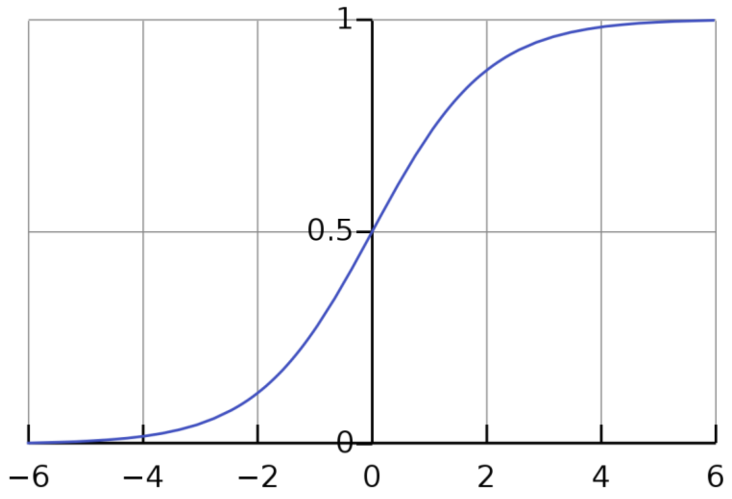
Fully Connected Layer: Convolutional and pooling layers break down the image into features and analyze them independently. The result of this process sends to a fully connected layer to get the final result.

Activation functions: The value of one layer's neuron cannot be fed to the next layer's neuron. Activation function helps the neuron to send it's result to the next layer.

Relu: f(z) = max(0, x)



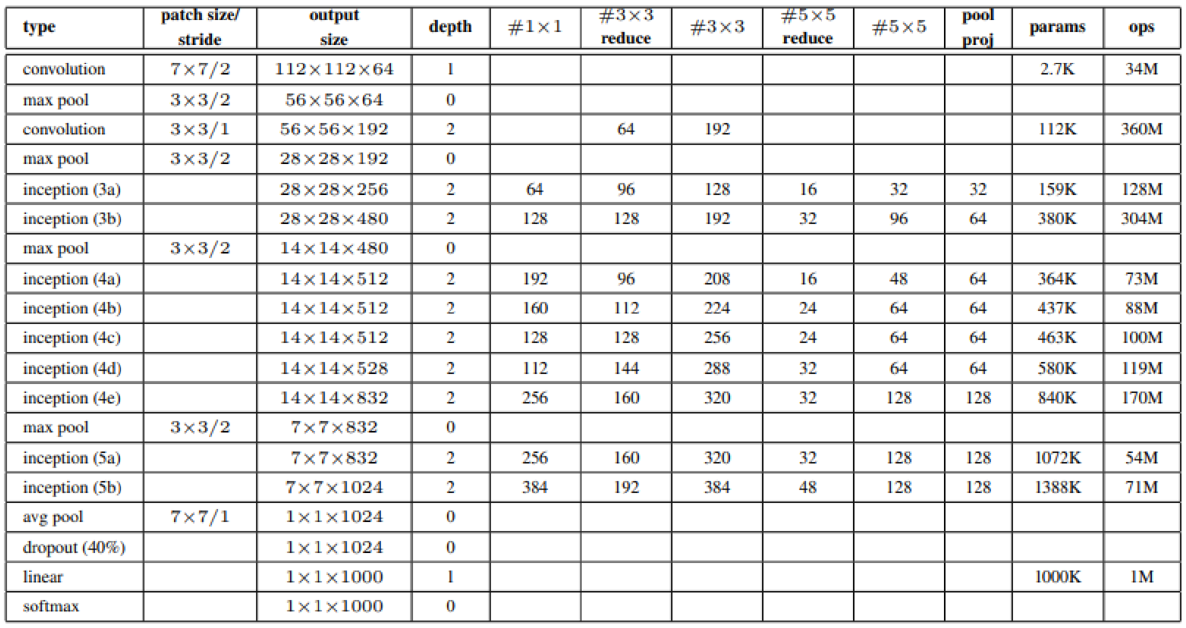
Sigmoid function:



accuracy: 0.9246 - recall: 0.9466 - precision: 0.9294 (after 5 epochs)

FUTURE: We have tried to build a basic CNN model. We have found that there are some models named Googlenet, Resnet, Alexnet, VGG16, VGG19. As we have got some ideas about the layers, we will try to implement Googlenet using the following link:

https://www.geeksforgeeks.org/understanding-googlenet-model-cnn-architecture/



type	patch size/ stride	output size	depth	#1×1	#3×3 reduce	#3×3	#5×5 reduce	#5×5	pool proj	params	ops
convolution	7×7/2	112×112×64	1							2.7K	34M
max pool	3×3/2	56×56×64	0								
convolution	3×3/1	56×56×192	2		64	192				112K	360M
max pool	3×3/2	28×28×192	0								
inception (3a)		28×28×256	2	64	96	128	16	32	32	159K	128M
inception (3b)		28×28×480	2	128	128	192	32	96	64	380K	304M
max pool	3×3/2	14×14×480	0								
inception (4a)		14×14×512	2	192	96	208	16	48	64	364K	73M
inception (4b)		14×14×512	2	160	112	224	24	64	64	437K	88M
inception (4c)		14×14×512	2	128	128	256	24	64	64	463K	100M
inception (4d)		14×14×528	2	112	144	288	32	64	64	580K	119M
inception (4e)		14×14×832	2	256	160	320	32	128	128	840K	170M
max pool	3×3/2	7×7×832	0								
inception (5a)		7×7×832	2	256	160	320	32	128	128	1072K	54M
inception (5b)		7×7×1024	2	384	192	384	48	128	128	1388K	71M
avg pool	7×7/1	1×1×1024	0								
dropout (40%)		1×1×1024	0								
linear		1×1×1000	1							1000K	1M
softmax		1×1×1000	0								