iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 10 **Issue:** XII **Month of publication:** December 2022

DOI: https://doi.org/10.22214/ijraset.2022.47984

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 10 Issue XII Dec 2022- Available at www.ijraset.com

Binary Image Classification using Deep Learning from Scratch

Saran Sri Dath Uppala[1], P. Sai Satya Murthy[2], SriGeethika Nagavajyula[3], A. Vamshi[4]
[1, 2, 3, 4]*Sreenidhi Institute of Science and Technology Yamnampet, Ghatkesar, Hyderabad, Telangana-501301.*

Abstract: ***Over the last several years, deep learning has seen a considerable increase in its usage and applications. One of the most basic algorithms in either machine learning or deep learning is image classification. Numerous businesses, like Microsoft and Google, use deep learning in their products, including search engines, Google Lenses, and other areas of technology. The enhancements and technical advancements in this area specifically serve to enhance the model's performance. The categorization of numerous photographs into two different groups is the focus of this study. In this study, the data is initially pre-processed to ensure that it can be easily inserted into the deep learning neural network model. Without using any pre-existing deep learning frameworks like Tensor Flow, the architecture and neural networks needed for this project are designed from the ground up. In this study, we essentially use deep learning neural networks that were built from scratch to attempt to discriminate between a cat and a dog. Each and every cost and activation function is calculated by hand. We build a two-layer neural network to divide the photos into two groups that meet our needs. For the training data, the developed model's accuracy is 98%, while for the test data, it is over 80%. If the data is suitably pre-processed, this deep learning model may also be used to classify other images.***

Keywords: Deep learning, Image, Classification, data, architecture, neural network.

I. INTRODUCTION

Classification of binary images is one of the most popular machine learning problems. The primary goal of this project's binary picture classification model is to determine if a given input image belongs to a certain class or not. Consider a binary image classification model, for instance, which labels a picture as either a cat or not based on its content. When a cat picture is fed into a deep learning model for binary classification, the model's output will be 1, indicating that the input image is a member of the cat class. If the model produces a result of 0, it means that the picture is not one of a cat [1].

The systems that are often used for picture categorization include machine learning. Machine vision and artificial intelligence software work together to provide amazing results for picture categorization ([2]. Making sure all photos are classified according to their distinct sectors or groups is the primary goal of image [3]. The many uses for image categorization technologies, including remote sensing, robot navigation, and vehicle navigation. Machine vision's lengthy history with image categorization makes it a significant problem today.

The task comprises a wide variety of intra-class pictures that are influenced by colour, size, context, and form. The existing models of binary image classifications are not efficient in the pre-processing of data. The dataset used by the existing models are very outdated and are inefficient for all the processes.

Also, the existing models use the Tensor Flow library and totally are dependent on the function and models that are in-built in the library that is imported. The proposed system is using the hdf5 datasets that re more efficient than the pre-existing and outdated datasets.

They are better in almost all the aspects when compared to other types of storing data. These advantages of the proposed system will be discussed further in the later topics. Using this method by developing a model from scratch, we can custom build the machine learning models that we require according to the hyper parameters and also gives us a lot of options. In this project, we try to develop a binary image classification model from scratch. We do not import any existing libraries or modules related to machine learning such as TensorFlow. The libraries that we import are only for data pre-processing such as NumPy, h5py and others that are required accordingly. The code is hand written for each and every function that we use to prepare this model. The forward and back propagation is also written in the form of code.

Even though, using the existing TensorFlow libraries make this problem a lot easier and can be done only in a few lines of code, but the main aim of this project is to implement hdf5 dataset and also to get a basic understanding of how a neural network model works.

II. RELATE WORKS

According to [4], a convolutional neural network-based image classification system was described in the journal CNN. By generating extra face pictures from the face image data, the training was done in a way that employed an equal number of face photos and non-facial images.

On the Face Detection Data Set and Benchmark (FDDB), the image classification system uses the biscale CNN with 120 trained data, and the auto-stage training achieves an 81.6% detection rate with only six false positives, while the state of the art achieves about 80% detection rate with 50 false positives.

The study used decision tree (DT) approaches for image categorization, according to [5]. Each hierarchical classifier in the DT is home to a number of datasets. To determine membership for each of the classes, it must be done. On the intermediate phases, the classifier permitted partial class rejection. This technique likewise had three phases, the first of which was locating the terminal nodes and the second of which was placing the class inside them. Partitioning the nodes is the third option. This approach is said to be extremely straightforward and very effective.

III. METHODOLOGY

In this project, for solving the binary image classification the activation function that we will be using is an RELU function that stands for Rectified Linear Unit Function. The Relu function is generally a combination of a Linear regression and a sigmoid operation.

Here as we are using matrices in the form of multi-dimensional arrays or lists, these types of data are considered to be vectors. Hence, the vector form of the linear regression function can be given as:

$$y = w(\text{transpose})x + b$$

Then the output obtained from the linear regression function is then sent as the input to the sigmoid function. The sigmoid function gives the output which denotes the probability whether the given input image belongs to a particular class or not.

These RGB values must be converted into a multi-dimensional. This entire process is done int the data pre-processing part. The test data images are converted into multi-dimensional lists and also the training images are converted into multi-dimensional images with their corresponding RGB values in each column and row of the array. Forward Propagation: Forward Propagation is the mechanism where it deals with generation of output for the given input. The data which is pre-processed in the previous module i.e., the data pre-processing phase, is given as input to the constructed deep learning model. Also, some of the hyper parameters such as weights for each input vector and the bias for each neural node are initialized to 0 at the starting. In forward propagation, let X be the input vector or input data. And W be the weight matrix and b be the bias. We need to pass the input vector or input data to an activation function to produce the output required. In this project, for solving the binary image classification the activation function that we will be using is an RELU function that stands for Rectified Linear Unit Function. The Relu function is generally a combination of a Linear regression and a sigmoid operation. The linear Regression is function that performs the summation of bias and the result of dot product of the input vector x and the weight matrix. The scalar equation for linear equation is generally given as:

$$y = wx + b$$

But here as we are using matrices in the form of multi-dimensional arrays or lists, these types of data are considered to be vectors. Hence, the vector form of the linear regression function can be given as:

$$y = w(\text{transpose})x + b$$

The next method that will be defining in our code is the optimize () function. The optimize function will be very helpful in calculating the updated values of the hyperparameters of the deep learning model which are weights and bias respectively. We usually update the weights and bias of a deep learning neural network which are our hyperparameters in this project using the learning rate and also the differential of the weights and bias which are dw and db that are calculated in the previous segment of the code. After calculating these values, we append the actual cost to the total cost that we get by calling the propagate () function that is previously discussed.

After implementation of the optimizing, we try to write the code for the prediction of whether the given input image belongs to a particular class or not (here in this case whether the given image is a cat or not a cat). The below screenshot represents the code that could be used for this task. If the output of the sigmoid function is above 0.5 then it would be rounded off to 1 denoting that the given image is a cat. If the output of the sigmoid function is less than 0.5 then it would be rounded off to 0 denoting that the given input image is not a cat.

The other and the last function that is to be implemented to completer our deep learning artificial neural network model is the model () method. This model () function wraps up the entire model and connects all the method that we have been discussing in the past few pages. It also initiates the process of prediction of the data by calling in the predict () method. Before that it also initializes the hyper parameters to zeroes by calling the other method that we have discussed earlier which is initialize_with_zeros () method

IV. IMPLEMENTATION ANALYSIS

The dataset for the binary image classification is usually a set of images that are either a cat or nor a cat that we will be using in this project. So, these images must be pre-processed and converted into the form that can be given as input to our deep learning neural network model. We all generally know that an image is combination of a number of pixels usually in thousands for a good quality image. So, in an image these pixels are arranged in a number of rows and columns. So, these images are stored in the form of three matrices corresponding to the red, green and blue values of each pixel the entry of each matrix as shown in the below image.

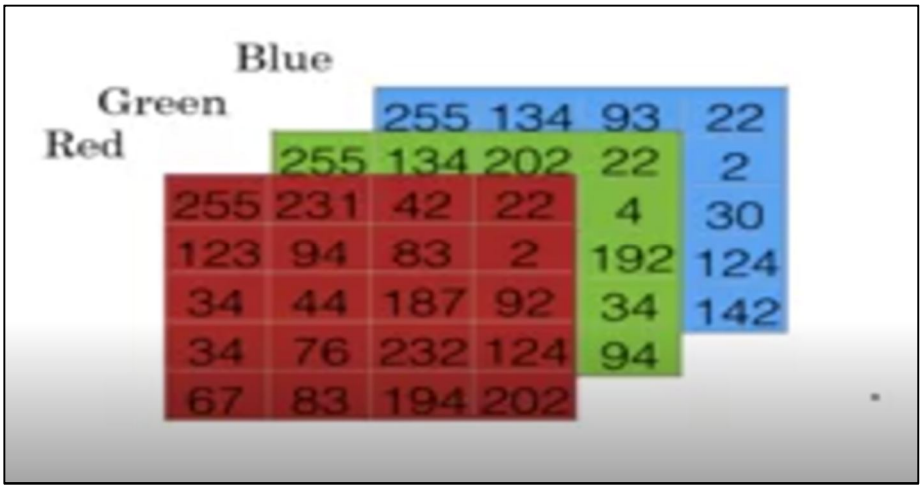


Figure 1 : RGB Matrices of an image

We can pass our own image after the entire model has been trained enough with the required accuracy. This can be done by importing the image and pre-processing the imported image as mentioned in the pre-processing module above. After preprocessing the image pass the pre-processed data into the model to get the evaluated output. Ger the returned values from the model and print the required output that is whether the given image belongs to a particular class or not.

The hyperparameters must be initialized to zero at the starting. To do this task we will be using a function named as initialize_with_zeroes to initialize the hyperparameters namely the weight matrix and the bias to zeroes .

The cost values for every 100 iterations that takes place inside the deep learning neural network model. Later at the end it displays the entire training dataset accuracy. After the training dataset accuracy, it applies the same model tot the test dataset and also displays the test dataset accuracy at last.

Train accuracy: 99%

Test accuracy:70%

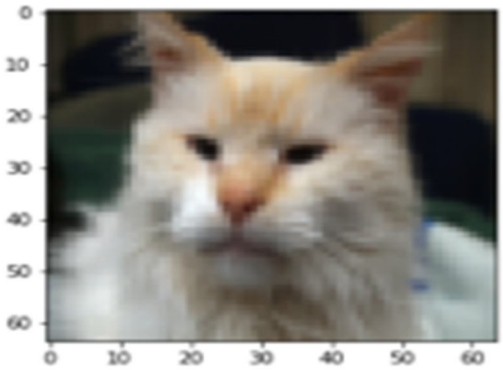


Fig 2 : Importing the dataset

Fig 3: Output for a cat image

Fig 4: Output for non-cat image

Fig 5 : Algorithm surviving in extreme conditions

Fig 6: Algorithm fails in rare conditions denoting the possible if improvement.

V. CONCLUSION

By constructing this Deep learning Neural Network, we have got to learn about how the internal system of a model works. In this project we have successfully construct a deep learning neural network that consists of 1 hidden layer. We have also implemented the Forward propagation using the Relu activation function and also implemented the Backward propagation using the gradient descent algorithm. The accuracy obtained after training the constructed neural network is about 70 percent which a decent start.

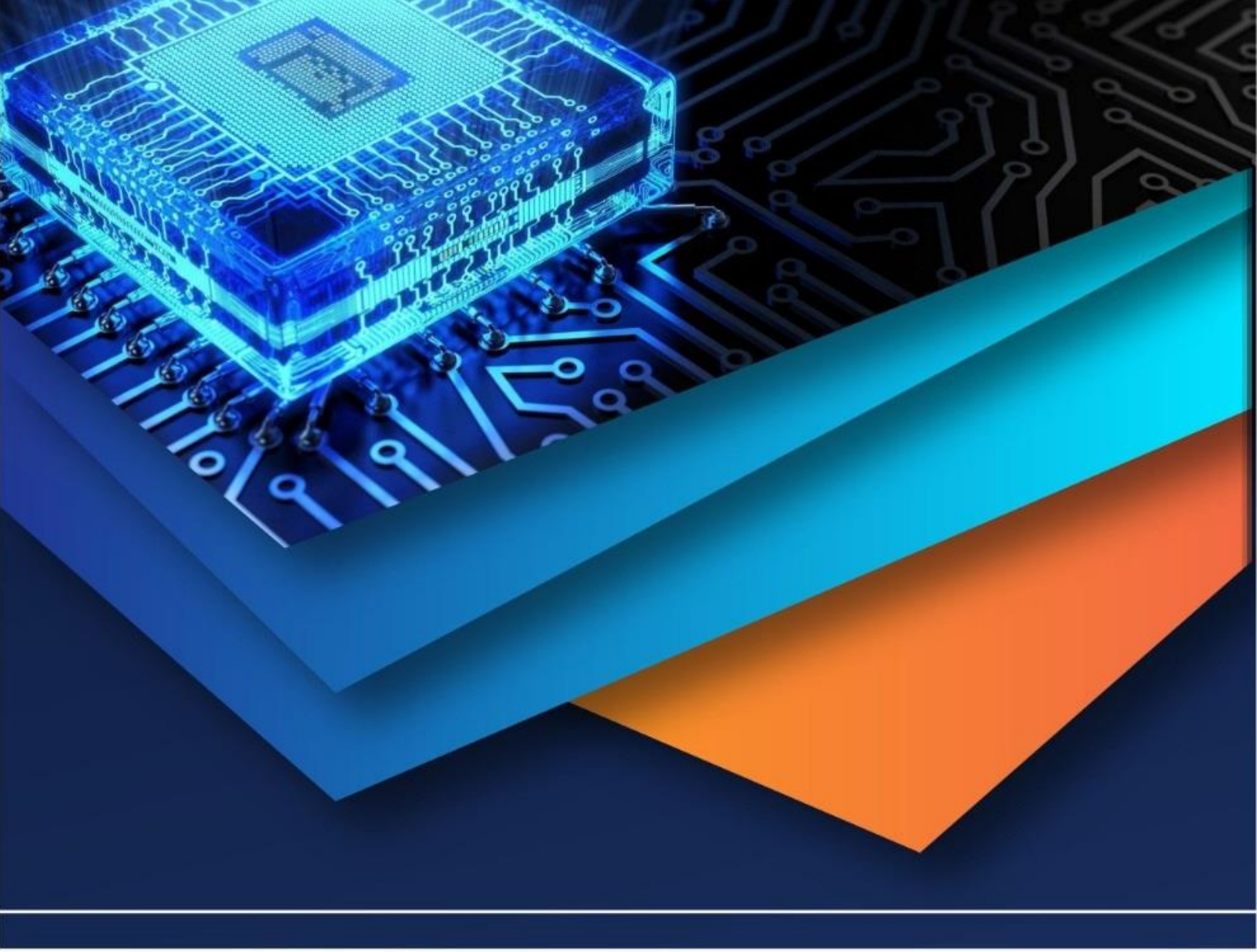
We have successfully implemented the deep learning neural network using the h5 datasets. This project successfully demonstrates the advantages of using the h5 datasets in our model. The efficiency of the model is up to great but there is scope for performance and accuracy improvements for the model we constructed in this project.

VI. FUTURE SCOPE

In future, this model can further be developed to increase the accuracy and dependency of the model. Right now, with the current generation we have been using only one hidden layer in our deep learning neural network. But we can improve the model, by adding a few more hidden layers to the deep learning model so that it can increase the overall accuracy of the model. Of course, adding a fer more hidden layers is going to reduce the overall simplicity of the deep learning neural network model. But it can be very helpful in being accurate. Also, we could be using other types of datasets such as bird vs non-bird. This helps to implement the deep learning neural network model that we have constructed in various real-life applications. We can also add some interface to create an app allow the user to interact with the model. In this way we will be able to construct a full and complete Deep Learning Neural network for solving the Binary Image Classification Problem

REFERENCES

[1] https://medium.com/techiepedia/binary-image-classifier-cnn-using-tensorflow-a3f5d6746697.
[2] Aizat Faiz Ramli, Hafiz Basarudin, Mohd Azlan Abu, Muhyi Yaakop, Mohamad Ismail Sulaiman, FUSA: Fuzzy Logic Based Clustering Protocol for Formation of Uniform Size Clusters, 2017 International Conference on Engineering Technology and Technopreneurship (ICE2T), pp 1-6, 2017.
[3] Gregor, K., Danihelka, I., Graves, A., Rezende, D. J., & Wierstra, D. (2015). DRAW: A Recurrent Neural Network For Image Generation. https://doi.org/10.1038/nature14236.
[4] Rastegari, M., Ordonez, V., Redmon, J., & Farhadi, A. (2016). XNOR-net: Imagenet classification using binary convolutional neural networks. Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 9908 LNCS, 525–542. https://doi.org/10.1007/978-3-319-46493-0_32.
[5] Sachchidanand Singh, Nirmala Singh "Object Classification to Analyze Medical Imaging Data using Deep Learning", International Conference on Innovations in information Embedded and Communication Systems (ICIIECS), ISBN 978-1-5090- 3295-2, pp. 1-4, 2017.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY