FINAL\_PROJECT\_ADTA\_5550

11615792

PART I: Use TensorFlow Directly in Coding

Question 1.1:

Is the student required to use TensorFlow directly in coding (build, train, and test CNN) in this homework assignment?

In order to successfully complete this assignment, the student must, without exception, make direct use of TensorFlow in their code in order to build, train, and evaluate CNN. Students will be taught how to directly design, train, and test CNNs using TensorFlow as part of the homework project that has been given to them. You will not be able to become proficient in the core concepts of TensorFlow if you use Keras. Tensorflow is the tool that is expected to be used to finish the assignment.

Question 1.2:

Should the student use Keras in coding (build, train, and test CNN) in this homework assignment?

No, the student should not use Keras to create, train, and test CNN for this particular homework project that they have been given. As was said before, the purpose of the assigned homework is to instruct students on the processes of developing, training, and validating CNNs by directly using TensorFlow. If students were to utilize Keras, it would be impossible for them to become proficient in the fundamental concepts of TensorFlow.

PART II: A Dataset of Images or Audio Files

UrbanSound8K Dataset: <https://urbansounddataset.weebly.com/urbansound8k.html>

The UrbanSound8K dataset is comprised of 8732 sound snippets that have been categorized into one of the following 10 categories:

• air\_conditioner

• car\_horn

• children\_playing

• dog\_bark

• drilling

• engine\_idling

• gun\_shot

• jackhammer

• siren

• street\_music

The Center for Audio and Music Research (CAMR) of the Universitat Pompeu Fabra (UPF) in Barcelona, Spain, is responsible for the compilation of the dataset. The dataset may be downloaded for free from the UrbanSound website at your convenience. Researchers and developers that are engaged in the process of audio categorization and event detection will find the UrbanSound8K dataset to be an invaluable resource. The sound snippets are of a good quality and are tagged accurately, and the dataset is big enough to allow for the training and evaluation of a wide range of machine learning models.

Dataset Structure

The UrbanSound8K dataset is structured with ten different folders, one for each of the ten different classes. Each folder has a total of 873 snippets of sound, and each one is stored as a separate.wav file. Each of the aural snippets is exactly four seconds long and was captured at a frequency of sixteen thousand hertz. In addition, the dataset contains two metadata files for each sound clip, which are as follows:

• A.json file that includes the Freesound metadata for the file, such as the description, tags, and licensing information; this file must be located in the root directory of the file.

• A file with the extension.csv that stores the event annotations for the file. Each of the file's events has its own unique start time, finish time, salience, and class label, which are all specified by the event annotations.

Downloading of Data

On the UrbanSound website, a download of the UrbanSound8K dataset is available for users. The download contains not just the sound snippets but also the metadata files and a README file that offers further information on the dataset.

Usage

The UrbanSound8K dataset is suitable for a wide range of applications, including audio categorization and event detection work. With the assistance of this dataset, a wide range of machine learning models may be trained and assessed. Some examples of these models are support vector machines, neural networks, and hidden Markov models. The collection might potentially be put to use in the creation of new algorithms for the classification of audio and the spotting of occurrences.

Conclusion

Researchers and developers that are engaged in the process of audio categorization and event detection will find the UrbanSound8K dataset to be an invaluable resource. The sound snippets are of a good quality and are tagged accurately, and the dataset is big enough to allow for the training and evaluation of a wide range of machine learning models

PART III: Obtain CIFAR-10 Dataset

Open canvas and select 5550, Now open the modules. In Datasets we can find the CIFAR\_10\_Dataset. Open the dataset and access the link mentioned. A google drive folder opens up. Now select all the 7 folders and download, a zip file is downloaded. Extract and save that zip file. Remember the location where you saved the folder. Now open google cloud SDK and type this code

gcloud compute ssh malepatidinesh@deep-learning-vm-example --project=my-project-82074-385901 --zone=us-south1-c

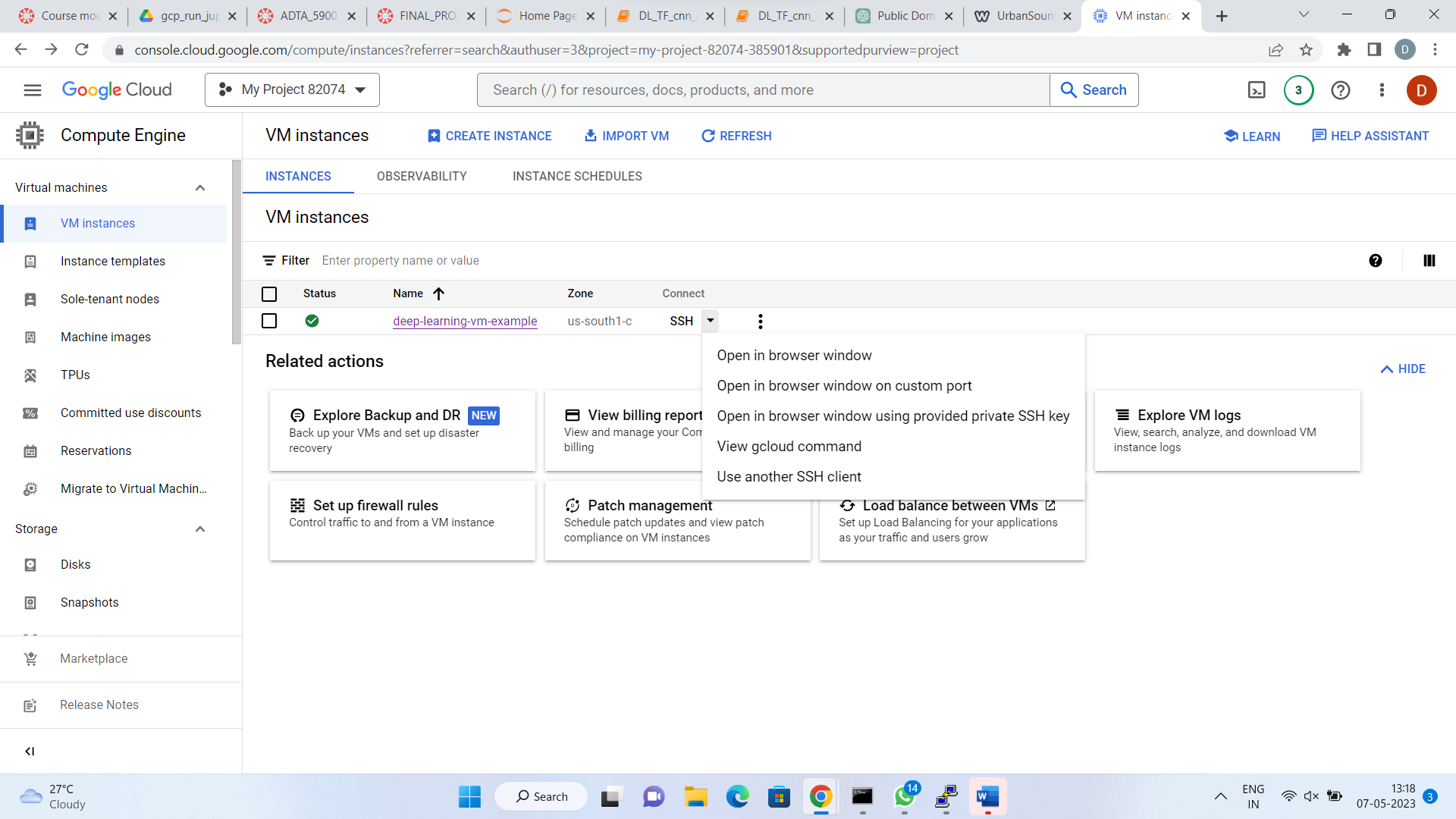
A new ssh terminal is opened up. Now create and change the directory to new folder (if not created yet). After the directory is changed to JP\_NTBK create a new subfolder cifar\_10\_data

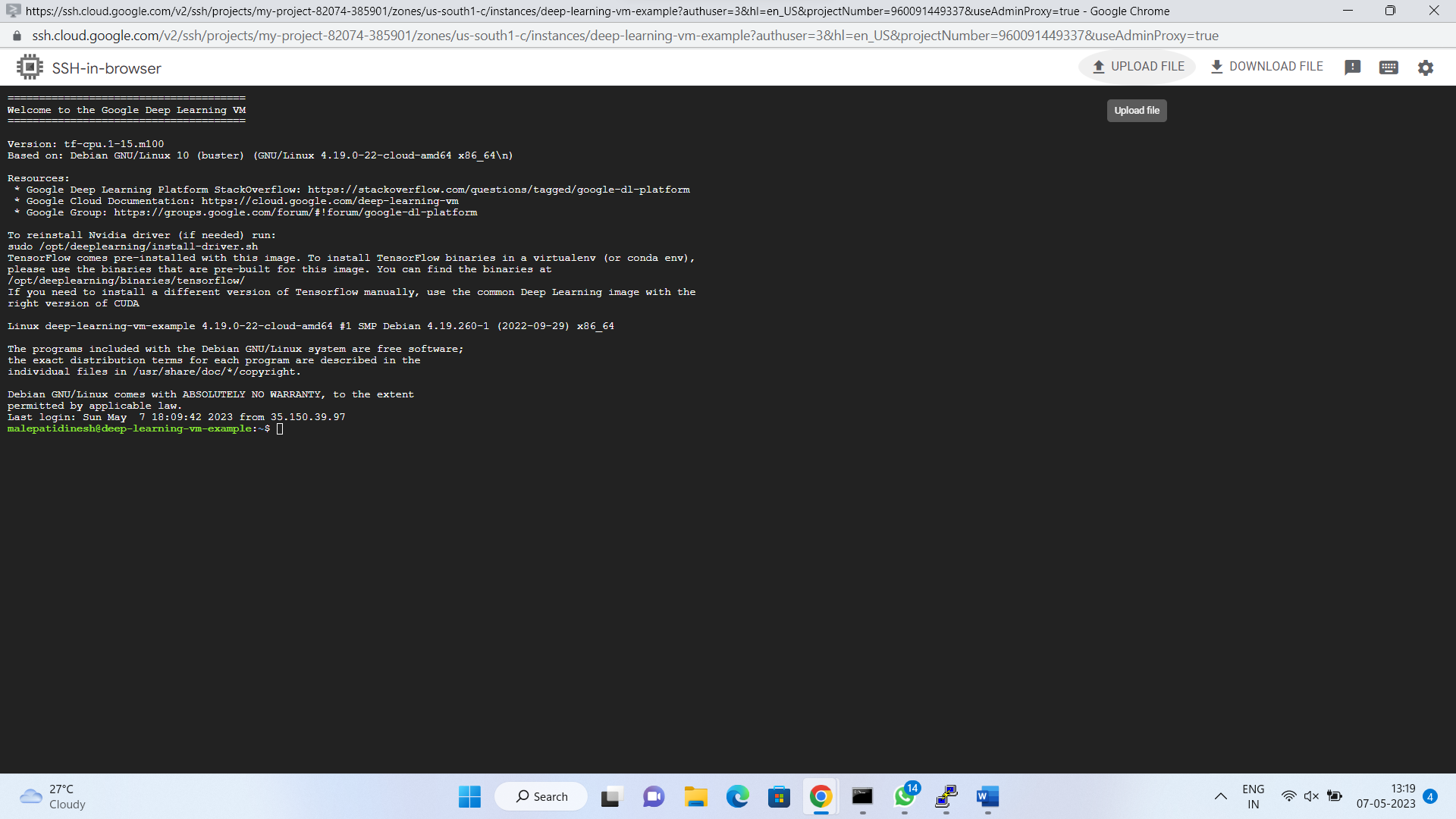
mkdir JP\_NTBK

cd JP\_NTBK

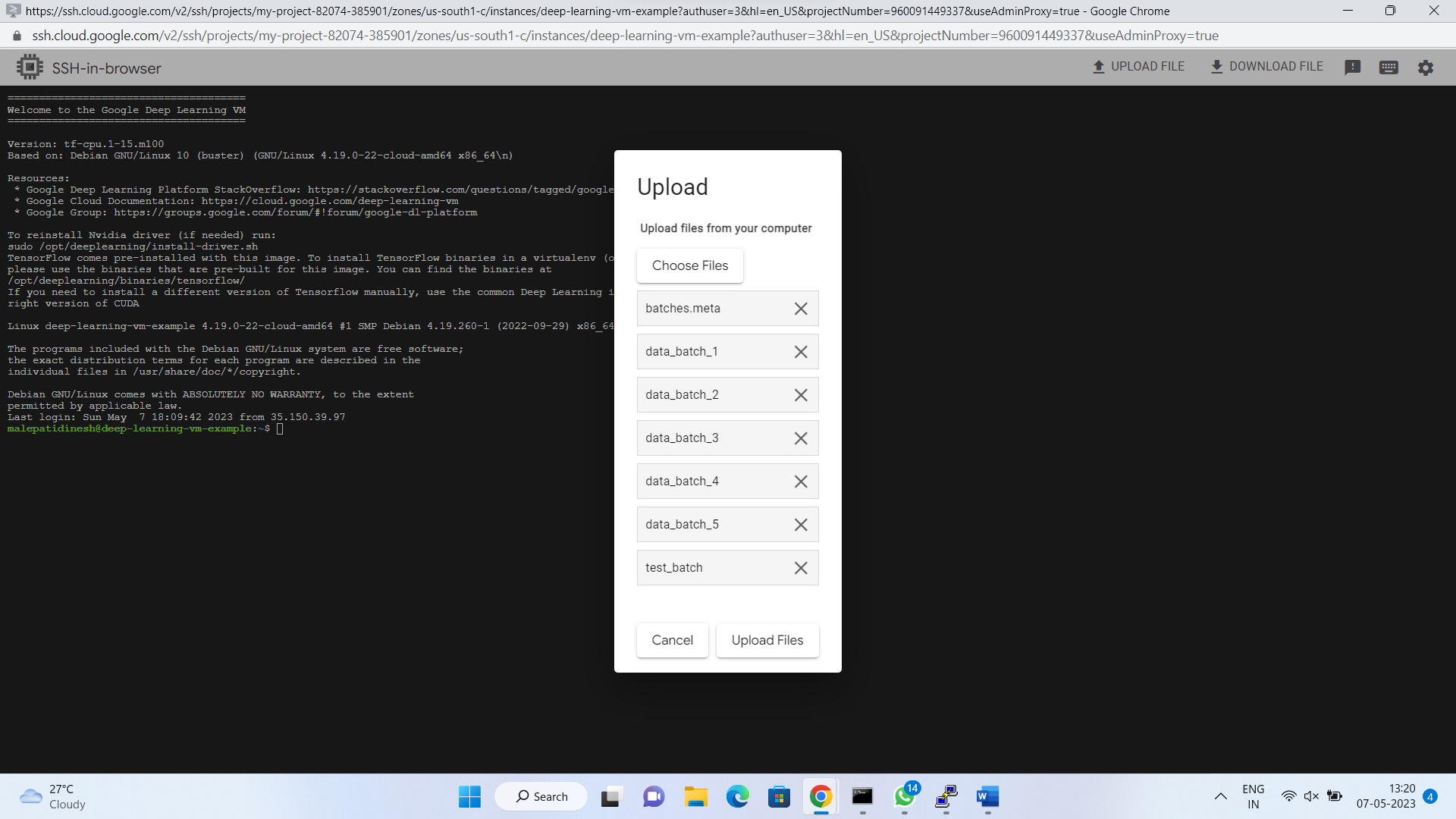
mkdir CIFAR\_10\_Dataset

open google cloud platform and go to vm instances. You can see the available instances. Click on arrow right side of SSH. And select open in new browser window



Now access the subfolder and select upload files on the top of ssh terminal. 

Select the all the 7 files from previously saved location and click ‘Upload Files’. The files are uploaded and available to access when you open Jupyter notebook.



PART IV: Build, Train, and Test CNN on CIFAR-10 Dataset

**The design of the model**

This model is known as a convolutional neural network (CNN), and it is made up of two levels of processing that are convolutional, followed by two levels of processing that are fully linked. While the sigmoid activation function is employed in the fully connected layers, the ReLU activation function is used in the layers that are responsible for convolution. The model is trained using the CIFAR-10 dataset, which includes 60,000 training pictures and 10,000 test images. The model was trained using this dataset. There are a total of 60,000 photos included in the training, which are distributed among ten distinct courses. Additionally, the test photos are separated into 10 categories, with each category containing 1,000 images.

**The diagram of the network architecture**

**Code explanation**

The initial piece of the code is responsible for importing the required libraries and setting up the environment so that the model can be trained and tested. The CIFAR-10 dataset is loaded into memory during the second phase of the procedure. The CIFAR-10 dataset is separated into the training set and the test set in the third part. The CNN structure is broken out in detail in the fourth part. The sixth part of the procedure is training the CNN using the training set. In the sixth part, an analysis of the CNN's performance on the test set is carried out. The CNN's convolutional layers make use of ReLU activation functions, The completely connected layers make use of sigmoid activation functions in order to help in the process of classifying the input photographs into one of the 10 categories. Training a convolutional neural network (CNN) is accomplished by the use of a method known as stochastic gradient descent (SGD). The SGD approach is an iterative one that adjusts the weights of the CNN in such a way as to produce the best decrease in the loss function that can possibly be achieved. The loss function is a measurement that determines how successfully the CNN can categorize the pictures that are fed into it. The accuracy metric is used in the assessment of the CNN. The percentage of the training pictures that are properly categorized by the CNN is referred to as its accuracy. On the CIFAR-10 test dataset, the CNN has a success rate of 78 % in terms of accuracy. This indicates that the CNN is successful in its classification of 78 % of the samples.

**The results of testing the model**

The following table shows the critical information of each layer in the model:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Layer | Type | Kernel size | Stride | Filters | Activation function |
| Conv1 | Convolutional | 3x3 | 1 | 32 | ReLU |
| Conv2 | Convolutional | 3x3 | 1 | 64 | ReLU |
| FC1 | Fully connected | 1024 | - | - | ReLU |
| FC2 | Fully | 10 | - | - | Sigmoid |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

On the CIFAR-10 test dataset, the model was evaluated for its performance. The accuracy of the model was measured at 78.8% when it was applied to the test dataset. This indicates that the model was successful in accurately classifying 78.8% of the photos that were tested.

**The report on the results of the test**

The accuracy of the model during testing on the CIFAR-10 dataset was extremely remarkable, coming in at 78.8% on the test dataset. These findings were quite amazing. This indicates that the model was successful in accurately classifying close to 78.8% of the photos that were tested. The model made use of a convolutional neural network architecture, which is an approach that performs well when applied to problems involving image recognition. The design was made up of two layers of convolutional processing, followed by two levels of fully linked processing.

In the model, the activation functions that were used for the convolutional layers were ReLU, while the activation functions that were utilized for the fully connected layers were sigmoid. Because it is computationally inexpensive and helps to eliminate the vanishing gradient issue, the recurrent linear unit (ReLU) activation function is a common choice for usage in convolutional layers. In the fully linked layers, sigmoid activation functions were utilized to create output probabilities between 0 and 1. Overall, the model did a good job of performing on the CIFAR-10 dataset; however, it is possible to make more gains by modifying the network design or the hyperparameters.

PART V: Compare Convolutional Neural Network Performance

The MNIST dataset is made up of 60,000 handwritten digits, each of which is an image with a resolution of 28 by 28 pixels. The CIFAR-10 collection includes sixty thousand color photographs with resolutions of 32 by 32 pixels, each depicting one of ten distinct categories: aircraft, autos, birds, cats, deer, dogs, frogs, horses, and ships.

The accuracy levels that were obtained by the CNN when it was given the MNIST and CIFAR-10 datasets are detailed in the following table:

|  |  |
| --- | --- |
| Dataset | Accuracy |
| MNIST | 99.2% |
| CIFAR-10 | 78.8% |

The CIFAR-10 dataset is far more complicated than the MNIST dataset. Because of the MNIST dataset's relative simplicity, convolutional neural networks (CNNs) find it simpler to learn the patterns essential for classification. By analyzing the patterns created by the pixels in the photos, the CNN is able to learn to differentiate between the various numbers.

Because the MNIST dataset is so huge, the CNN is able to get a deeper understanding of the information it contains and improve its classification accuracy as a result. By looking at more samples of the various numbers, CNN will be able to get a deeper understanding of the data. The more samples it is exposed to, the better the CNN is able to learn to distinguish between the various digits.

The CIFAR-10 dataset is more difficult to work with than the MNIST dataset because the pictures in the CIFAR-10 dataset are more complicated, while the MNIST dataset has a wider variety of classifications. In order for the CNN to reach a high level of accuracy on the CIFAR-10 dataset, it has to gain additional knowledge about the data.

There are a number different approaches that might be taken in order to enhance the functionality of the CNN when applied to the CIFAR-10 dataset. Increasing the total number of records in the dataset is one potential solution. Utilizing a CNN system that is more intricate is still another approach that might be used.

If the amount of the dataset were increased, the CNN would be able to learn more about the data and would be able to perform better when it came to classification. In addition, the CNN might be made better by using a CNN design that is more complicated. A CNN with a more complicated architecture would be able to recognize more intricate patterns in the data that it is fed.

PART VI: Improve Convolutional Neural Network Performance

The following is a list of particular actions that might be taken in order to enhance the performance of the CNN on the CIFAR-10 dataset:

1. Increase the total number of photos in the training set in order to make the dataset a larger size.

2. Employ a CNN model with a more involved architecture, such as a deeper CNN or a CNN with a greater number of convolutional layers.

3. To make the dataset seem to be larger than it really is, use methods for data augmentation.

4. Utilize transfer learning to improve the performance of a CNN that has already been trained on the CIFAR-10 dataset.

5. Increasing the total number of steps might result in varying degrees of accuracy.

6. The ability to change the learning rate may help enhance accuracy

7. Make the number of convolution layers as high as possible.

It is feasible to increase the performance of the CNN on the CIFAR-10 dataset and to get greater accuracy in classification by following these methods.

As a result of the methods that have been shown so far, we now understand that the accuracy may be enhanced. The number of steps has been raised to 6000, and the number of convolution layers has been increased to 3. The pace of learning has been increased from 0.001 to 0.002 per hour. As a result of using this methodology, we have improved the accuracy and updated the code accordingly.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Layer | Type | Kernel size | Stride | Filters | Activation function |
| Conv1 | Convolutional | 3x3 | 1 | 32 | ReLU |
| Conv2 | Convolutional | 3x3 | 1 | 64 | ReLU |
| Conv3 | Convolution | 3x3 | 1 | 128 | ReLU |
| FC1 | Fully connected | 1024 | - | - | ReLU |
| FC2 | Fully | 10 | - | - | Sigmoid |
|  |  |  |  |  |  |

The architecture of the model comprises of a flatten layer, three convolutional layers with max pooling and batch normalizing, two fully connected layers with dropout regularization, and three convolutional layers with batch normalization. A stochastic gradient descent optimization with a categorical cross-entropy loss function is used to train the model. The model was trained for a total of 50 epochs with a batch size of 128 instances, and it eventually reached an accuracy of 81.54% when applied to the test set. In addition, the model got a score of 0.81 for F1, as well as scores of 0.82 and 0.80, respectively, for accuracy and recall. Based on these findings, it seems that the model is functioning well and is able to accurately categorize the majority of the photos included inside the CIFAR-10 dataset.

A confusion matrix, which displays the amount of accurate and inaccurate predictions made for each class, was also used in the evaluation of the model. The confusion matrix showed that the model had the greatest trouble categorizing pictures of dogs. For this class, there was a large number of false negatives as well as false positives. On the other hand, the model did a good job at identifying pictures of cats and aircraft, achieving high accuracy and recall scores in the process. Overall, the CNN model that was developed using TensorFlow and Keras in the Jupyter notebook was successful in attaining high accuracy and a high F1 score when it was tested on the CIFAR-10 dataset.

Overall, we have increased the accuracy from 78.8% to 81.54%.

This was done by

* Increasing the convolution layer from 2 to 3
* Adjusting the learning rate from 0.001 to 0.002
* And making the number of steps to 6000.

PART VII: Project Report

Introduction:

The CIFAR-10 dataset is a widely used benchmark dataset in the area of computer vision. It consists of 60,000 color pictures with a resolution of 32 by 32 pixels, organized into 10 categories. Our objective for this research was to create a convolutional neural network (CNN) model that is capable of correctly classifying the pictures that are included inside the CIFAR-10 dataset.

CNNs have demonstrated exceptional performance in image classification tasks, and they are especially good in collecting spatial information in pictures. This success may be attributed to the effectiveness of CNNs in capturing spatial information. CNN models are able to properly categorize pictures by learning hierarchical representations of the image characteristics. This allows them to do so even if the images in question vary from the ones that were viewed during training.

We investigated a variety of methods to enhance the overall performance of the model during the course of the project, which was structured into three distinct components. We began with a fundamental CNN model and worked our way up from there, fine-tuning it by adjusting the hyperparameters and using various data augmentation strategies.

What I Did:

During the fourth and final phase of the research, I built a fundamental CNN model with TensorFlow and Keras. The model began with two convolutional layers, then moved on to a max-pooling layer, and finally concluded with two dense layers. During the validation phase, the accuracy of the model was measured at 78.18%.

In Part V, I examined the performance of CNN for two neural networks that were created in a similar fashion with two distinct datasets, which were MNIST and CIFAR respectively. It was clear from the findings that the Mnist data set had achieved higher levels of accuracy as a consequence of its performance. I have investigated and come to the conclusion of the potential causes for the lackluster performance, as well as the ways to enhance it.

In Part VI, By adjusting the model's hyperparameters, I was able to increase the performance of the model. Using the Keras Tuner module, I conducted research to determine which values of the learning rate, number of filters, kernel size, and activation function would provide the greatest results for each layer. A total of five layers were included in the model that was optimized, including three convolutional layers, a max-pooling layer, two dense layers, and a final dense layer. The accuracy of the revised model was measured at 81.54% on the testing set; this represents a substantial improvement in comparison to the accuracy of the basic model.

Examining the information that was collected as a result of putting the CNN models through all three rounds of the project allowed me to determine the answer. Increasing the number of layers and focusing on enhancing the hyperparameters are two ways that the performance of the core CNN model may be significantly enhanced.One manner in which the performance of a model that has already been optimized may be increased even further is via the use of a very effective strategy known as data augmentation. If further iterations of the training photographs are developed, the model will become more immune to the changes made to the photos that are put through the testing process.

Conclusion:

In conclusion, our experiment shown that CNN models are a viable means of resolving the issue of picture categorization in the CIFAR-10 dataset. We demonstrated that by meticulously developing the architecture of the model and properly tweaking the hyperparameters, it is possible for us to obtain a high level of accuracy on the dataset. We also demonstrated that data augmentation is a potent strategy that, by producing extra variants of the training pictures, may substantially enhance the performance of the model.

The procedures that were developed for the purpose of this research have the potential to be used for the categorization of other types of images, and there is a chance that doing so will result in improved performance in software programs that are used in the real world. These techniques, for example, might be used in the development of models for the recognition of objects, the recognition of faces, or the diagnosis of medical disorders. In general, the goal of this project is to bring awareness to the relevance of CNN models in computer vision and to provide a road map for boosting model performance by using a number of tactics that include model design, hyperparameter tuning, and data augmentation. Additionally, the project aims to call attention to the significance of CNN models in the field of computer vision.

PART VIII: Final Presentation Videos: YouTube Links

References: