Project

Description

Of

Facial Expression

Recognition

Using

CNN

Contents

1. Data Gathering-------------------------------------03
2. Data Processing------------------------------------04
3. Data Visualization---------------------------------06
4. Feature set and Target Value Preparation----07
5. Model Generation and Fitting-------------------08
6. Model Evaluation-----------------------------------09
7. Model Deployment--------------------------------10

(2)

Data gathering

As we have done the whole project on the Google Collab Notebook we had to gather the data from the source ( <https://www.kaggle.com/ashishpatel26/facial-expression-recognitionferchallenge> ).

So we can download the data on the local machine and upload it to Collab or we can take the API token from Kaggle and bypass the download which will be faster and convenient method which actually has been done.

1. Changed the Google Collab runtime to GPU as we are doing works with CNN, CPU may work slower, so changing to GPU is a much optimized approach.
2. Took the Kaggle API token (should be named as *‘kaggle.json’*) and uploaded that on Google Collab.
3. Then written some code which helps to connect Collab with Kaggle and secured the connection.
4. Then pasted the API token of the dataset and thus the data downloaded as a .zip file.
5. As the data is in .zip file I had to extract the data into main folder.

And thus we have gathered the data.

Primary Visualization

After gathering the data I had taken it into a dataframe to process the data further.

It can be seen that there were a README file, a .bib file, the main data file in .csv format and a sample submission file in .csv also.

I had seen that the main data has 3 columns, first one holding the emotion types, second one holding the image pixel which are stored in a linear way and the third column holding the use of data (e.g. train data).

I also found that the data has no leakage and the train data size is 28709 and the public and private test size is 3589.

The emotions are not well tuned to each other as they are quite varied (the disgust facial expression is very less than any other emotion in all datasets).

Data Processing

The data is not quite prepare the feature set and target value. So I had to deal with some steps –

1. I had to take the pixel columns and prepare it in tabular format. As the values are in between 0 to 255 I had to divide all the values by 255 to get float type values which will help in data visualization.
2. The next step I took is to connect this prepared dataframe of pixels with the main dataframe.
3. Then split the dataframes into train, validation and test. The public test has been taken for validation and the private test as test data.
4. Then deleted the feature named ‘Usage’.

Data Visualization

The first work is to see the target value distributions in the datasets.

Observation indicates that some emotions are in a very large amount and some are in negligible amount. Still we cannot drop any target value as they do present in the test and validation and will be checked for accuracy.

There is one way to diminish this issue to take all the type of emotions in same counts.

But this makes the dataframes very small to train on the NN and that would not be a good approach.

So I left as it was.

So the dataframe now contains only the pixels and the emotion feature. After dropping the emotion we have the dataframe which has 2304 features.

So, it is thought that the image might be in square shape and the shape of the image height would be = 48.

So after reshaping one row of pixel we really got the picture and that was in correct orientation.

After visualization we made an array as a decoder which told what picture hold what emotion.

Then I visualized every type of emotions.

Feature set and Target Value Preparation

As we are preparing the feature set to feed into the NN model we have to convert the data into numpy arrays as this is a better approach to generate higher dimensional tensors in the model.

As we are using ConvNet to train the data so we have to make the data into 2d tensors. So we have to reshape the data into actual image format which is (48, 48).

As the target values are in integer format so we don’t have to convert their data type.

Model Generation & Fitting

The model for this project is a custom “**Sequential Model**”.

On that there are several Convolutional layers, Dense Layers and some Dropout of Neuron.

Thus we have prepared a 4-5 type of custom sequential model and checked those with different batch size and epochs.

We achieved over fitted condition in the first models and under fitted on the next models.

After increasing the batch size and the epoch of the last model gave us promising accuracy.

I used basically 2 basic structure of sequential model

* A:

Conv2D 🡪Dense 🡪Dropout 🡪 Dense 🡪Dropout 🡪 Flatten 🡪 Dense 🡪 Final Output

* B:

Conv2D 🡪 Conv2D 🡪 MaxPool2D 🡪 Dropout 🡪 Conv2D 🡪 Conv2D 🡪 MaxPool2D 🡪 Dropout 🡪Flatten 🡪 Dense 🡪 Final Output (Type B is the promising model)

Model evaluation

Model evaluation metric: accuracy score

Final model reached accuracy:

1. Train: 0.6605
2. Validation(Private Test): 0.5804
3. Test(Private Test): 0.5887

Model Loss: Sparse Categorical Cross entropy

1. Train: 0.8757
2. Validation(Private Test): 1.1807

Model Deployment

After evaluating the model we have now come to main phase of the whole project.

We have to submit the project as the submission file.

So I converted the prediction from the test data and created the dataframe and made an output in .csv format.

(Submission file named as ***Final\_submission.csv)***

Thank

You

Project Submitted By

Sagnik Roy

IT: 2019-2023

Jalpaiguri Government engineering College