Capstone Project

Face Emotion Recogniton

By - Shourya Chandra Sai

Project Agenda

1. **Notebook Segment**

- Data description
- Defining the problem statement
- Data pipeline & Data-Preprocessing
- Convolutional neural network
- Saving & downloading model architecture & weights

2. **Capstone 5 Project Directory Segment**

- Virtual environment creation
- Git initialization
- Camera.py script
- main.py script
- License, Requirements.txt, Readme files
- Deploying the model

3. **Conclusion**

Notebook Segment

Data Description:

Below is the description of all the columns in the dataset.

The name of the data set is fer2013 which is an open-source data set that was made publicly available for a Kaggle competition. It contains 48 X 48-pixel grayscale images of the face. There are seven categories (0=Angry, 1=Disgust, 2=Fear, 3=Happy, 4=Sad, 5=Surprise, 6=Neutral) present in the data. The CSV file contains three columns

1.***emotion*** that contains numeric code from 0-6 and

2.***pixel*** column that includes a string surrounded in quotes for each image.

3.***Usage*** column that lets us know what row should be used for training, validation & testing.

Defining the problem statement:

In this project, We try to detect the emotions of a student in order to understand whether that student is paying attention or not.

Data Pipeline & Data-Preprocessing:

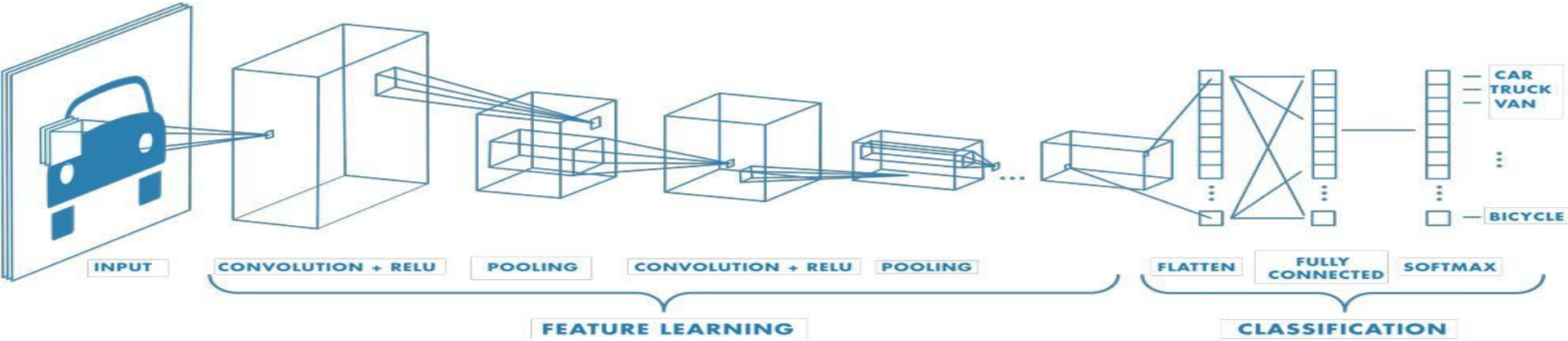
Under this sub-segment, I have performed the following pre-preprocessing procedures:

Under the usage column, there were 3 unique values viz. Training, Public Test, Private Test.I I have used Training data to train the model and Public Test for testing the model performance. I did not use Private Test data for validation purposes because of the limited CPU, and GPU that I have. Also changed the shape of X_data(X_train, X_test) into m(number of examples) * *48(pixel size of the image)* * 48(pixel size of the image) * *1(grayscale). Also performed one-hot encoding on Y_data(Y_train, Y_test) resulting in m(number of examples)* * 7(number of classes) shape.

What is CNN?

These layers perform operations that alter the data with the intent of learning features specific to the data. Three of the most common layers are convolution, activation or ReLU, and pooling.

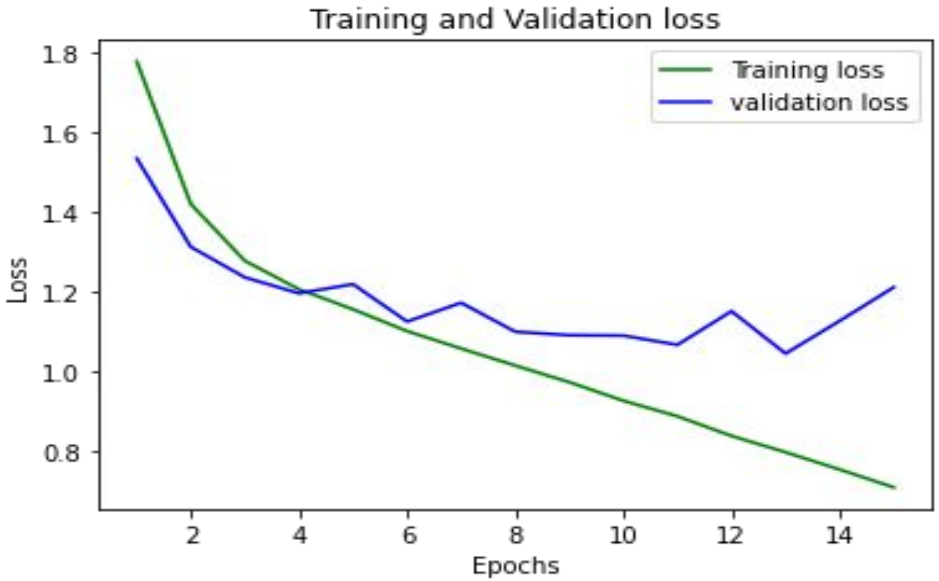
- **Convolution** puts the input images through a set of convolutional filters, each of which activates certain features from the images.
- **Rectified linear unit (ReLU)** allows for faster and more effective training by mapping negative values to zero and maintaining positive values. This is sometimes referred to as *activation*, because only the activated features are carried forward into the next layer.
- **Pooling** simplifies the output by performing nonlinear downsampling, reducing the number of parameters that the network needs to learn.
- These operations are repeated over tens or hundreds of layers, with each layer learning to identify different features.

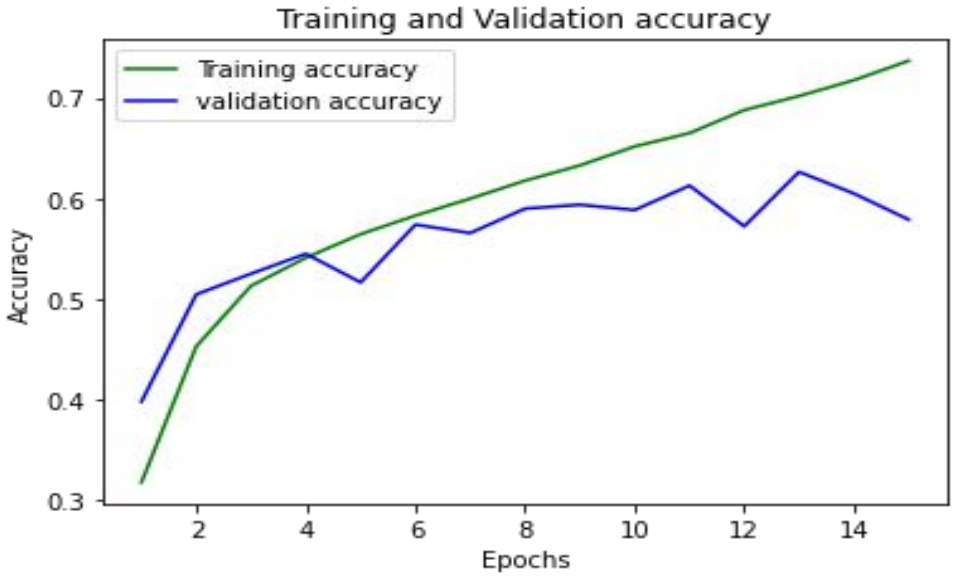


Convolutional neural network:

The model was inspired by Goodfellow, I.J., et.al. (2013) paper. Challenged in representation learning: A report of three machine learning contests. The model has 4 Conv blocks. Each Conv Block includes the following Conv2D, Batch Normalization, Activation, MaxPooling2D, and Dropout. It also has 2 Dense layers. Each Dense layer is followed by Batch Normalization, Activation, and Dropout. For compiling the model Adam was used as an optimizer, Categorical binary cross entropy was used as the loss function and accuracy was used as a metric. The model had a training accuracy of ~= 73% and testing/validation accuracy of ~= 60%.

The general trend of losses on both the datasets are in reducing fashion and the general trend of accuracies on both the datasets are in increasing fashion with the increase in the number of epochs.





Saving & downloading model architecture & weights:

I saved the model weights & architecture in an h5 & JSON file respectively. These saved values will be later used for making predictions.

Capstone 5 Project Directory Segment

Virtual environment creation:

I have created a virtual environment inside the project directory. Virtualenv is used to manage Python packages for different projects.

Using virtualenv allows you to avoid installing Python packages globally which could break system tools or other projects.

Git initialization:

I have also initialized git inside the project directory to version control the project.

Model.py script:

Model.py file was created to load the model's architecture and weights and to also make emotion predictions.

Camera.py script:

Camera.py file was created to identify a person's face from the webcam, create a bounding box around the person's face, and also predict emotions with the help of Model.py script.

main.py script:

main.py file is used to create a flask web application with the help of Model.py, Camera.py scripts, and index.html file.

Note: I have used flask and not streamlit because the stream-webrtc library doesn't work properly on Mac M1. I have informed the same to the doubt resolution experts and they have asked me to serve it on the flask.

License, Requirements.txt, and Readme files:

I have used an MIT license for the project.

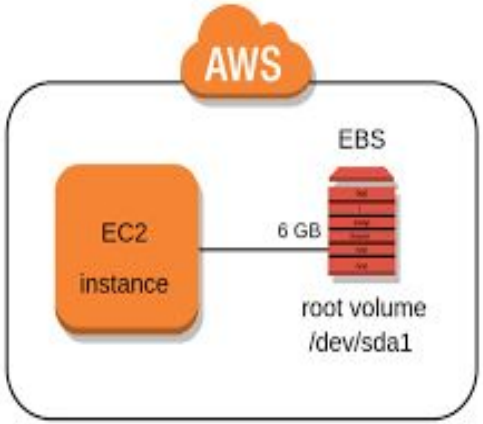
The Requirements.txt file includes all the dependencies of this project.

The Readme File specifies the procedure for running the application.

What is AWS EC2?

Amazon Elastic Compute Cloud (**EC2**) is a part of Amazon.com's cloud-computing platform,

users to rent virtual computers on which to run their own computer applications



Deploying the model:

I have deployed the model on the AWS EC2 ubuntu server by creating an instance of the server, connecting to the server from the terminal using the .pem key file, installing python, pip, and all the dependencies in the requirements.txt file using various commands and running the main.py file using ‘python main’.py command

Note:I have deployed the model on a server but the server can’t access the webcam of your local machine resulting in a scenario where the website works but the camera can’t be opened. I thought of deploying the model on Heruko but on Nov 28th 2022 Heruko ended the free subscription. So in order to show the working of the model I recorded a video of myself running the app which is present in the google drive folder.

Conclusion

- Using the dataset from kaggle we were able to build a CNN model which is capable of recognizing the facial emotion of a user.
- Pre-processed the data by segmenting the data into X_train,X_test,y_train,y_test with the help of the Usage column in the dataset.Also changed the shape of X_data(X_train,X_test) into m(number of examples) * *48(pixel size of the image)* * 48(pixel size of the image) * *1(gray scale). Also performed one-hot encoding on Y_data(Y_train,Y_test) resulting into m(number of examples)* * 7(number of classes) shape.
- The model obtained 74% accuracy on training set and 58-63% accuracy on test set. Plotted a line chart of training and testing losses as well as training and testing accuracy. The general trend of losses on both the datasets are in reducing fashion and the general trend of accuracies on both the datasets are in increasing fashion with the increase in the number of epochs.
- We saved the model weights & artitecture in a h5 & json file respectively.
- Created a project directory, created a virtual environment and initialized git for versioning of the code
- Written scripts like model.py, camera.py, main.py. Each script is used to solve different problem.
- Finally, Deployed the model on Ubuntu server using AWS EC2 service.

Dependencies

The following are the dependencies for the project:

Thank you.