PYTHON & DEEP LEARNING PROJECT REPORT

FACIAL EMOTION CLASSIFICATION

Keerti Bajaj
Srikanth Jeniga
Anjali Bodduluri
Rishi Dhara

Course Coordinator - Dr. Yugyung Lee
Supervisor - Prof. Ahmed Albishri

Submitted to:

School of Computing and Engineering
University of Missouri - Kansas City

Abstract

After the impact of Covid-19 pandemic, the importance of emotional well-being came into the highlights. Mental health is an important aspect of human life cycle that effects our day-to-day activities and social interactions. Depression and Anxiety are impacting many lives. When someone is not at mental peace, even a single act of asking and caring can mean a lot and in extreme cases, can save a life as well. A classifier that can predict the emotional state of a person by accessing a snapshot of their facial expression can help us better understand their state of mind. The primary objective of the project is to support mental well-being. Today's competitive world bring a lot of pressure to people of almost all ages. One or the other reason exits that makes us feel anxious and might lead to severe consequences. Having someone to determine and understand emotions can help many people overcome their fears and help them to be able to accept and tackle hard times.

Introduction

Deep Learning is a form of machine learning and artificial intelligence that is inspired by the structure of human brain. The stacked hierarchy of deep learning algorithms help to build models where traditional machine learning models fail. An image classification model is a deep learning model trained to identify the class of the image. Use of Convolutional Neural Networks for image classification helps in extracting the underlying features of an image. There are various pre-trained models such as VGG-19, GoogLeNet, ResNet34, EfficientNet. These models use convolutional layers for image classification.

ResNet34 has thirty-three convolution layers and one fully connected layer. FastAI is a library that provides high level components and pretrained models for various applications. The learning in fastAI includes high learning rate which makes model training faster but other optimizers are included for the model to converge.

Numpy, Pandas, Sci-kit Learn, OpenCV are some of the libraries used for processing the image data or csv data. Tensorflow and keras are widely used where layers are stacked to build a deep neural network. Keras offers various evaluation metrics to validate our models and optimize them better.

Flask is a Python "micro-framework" based on WGSI toolkit that helps developers to control how the user data is accessed. It provides the tools, libraries and methods to build a web application and deploy our model .

Data Collection & Pre-processing

The dataset that contains various images human facial expression is available on Kaggle. There are seven classes for these images: Happy, Sad, Fear, Surprise, Neutral, Angry and Disgust. Each image belonging to a particular class is stored in a folder named after the class. The data-set available is already preprocessed. Each image available has a original size of (48 x 48).

OS library helps in reading data from the disk. OpenCV helps in loading and transforming images. For the images to be labelled correctly, loop over the directories of the data-set and append each loaded image and it's corresponding label to a list. The images are re-scaled to (100 x 100). There are a total of 25,928 RGB images, 3704 for each class. For an image, each value is between 0 to 255, which is normalized to a value between 0 and 1. Class labels are one-hot encoded since the model cannot process non-numeric data. The lists of data are then converted to numpy arrays.

Model

The model built using Keras has two convolution layers and two dense layers. The convolution layers are followed by MaxPooling layers to reduce the spatial dimensions. Adam optimizer uses both adaptive gradient descent and RMSProp for training of the network.

Early stopping prevents over-training of the data. Reducing learning rate when training doesn't improve significantly over some epochs enhances the model. These are the two callbacks used to fit the model.

From the data, 20% is used for validating the model for unseen data. The model is trained for 30 epochs. During training, model metrics are stored in a variable and accessed to plot accuracy and loss for training and validation. This model is saved in .h5 format which is further used for deployment on the web application.

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 98, 98, 200)       5600
_________________________________________________________________
activation_1 (Activation)    (None, 98, 98, 200)       0
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 49, 49, 200)       0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 47, 47, 100)       180100
_________________________________________________________________
activation_2 (Activation)    (None, 47, 47, 100)       0
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 23, 23, 100)       0
_________________________________________________________________
flatten_1 (Flatten)          (None, 52900)             0
_________________________________________________________________
dropout_1 (Dropout)          (None, 52900)             0
_________________________________________________________________
dense_1 (Dense)              (None, 50)                2645050
_________________________________________________________________
dense_2 (Dense)              (None, 7)                 357
=================================================================
Total params: 2,831,107
Trainable params: 2,831,107
Non-trainable params: 0
_________________________________________________________________
```

Figure 1: Summary of the Model

The model training took about 25 - 30 minutes for each epoch to train. As a back up another model is built using transfer learning from ResNet34 using FastAI.

Web Application

A basic HTML page is designed to upload an image and predict the class of the image. Flask is used to manage the model server and is scripted in python language. The image input from the page is preprocessed using OpenCV. The saved model from training is loaded into this python file and is used for prediction.

The web application has an image upload input option. Once the image is uploaded, it displays the image. Once the predict button is clicked, an event is called to make predictions and the probability of each class is displayed.

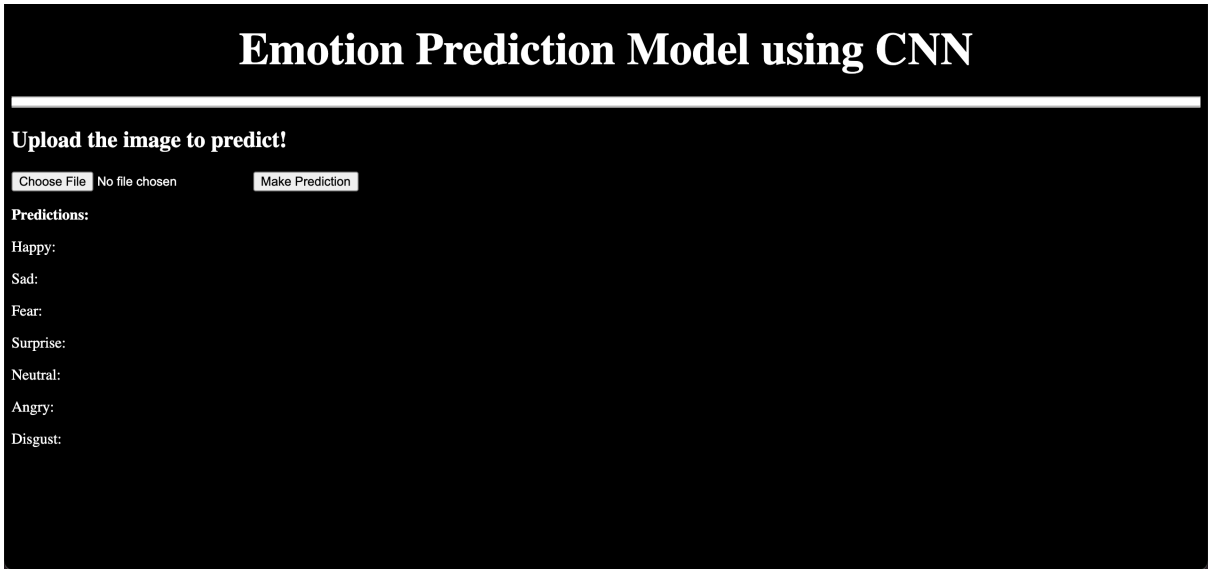

Figure 2: Web Application Preview

Challenges Faced

Due to the huge size of the data and hardware limitations, the training time for the model is too high. When Google Colab was used for overcoming the hardware limitations, an unknown error cause only about 600 images to be used for training the model. This raised a question of not knowing which images were used to train the model and was it enough for the classification of seven classes.

The ambiguity in the facial expressions and resemblance of some of the emotions with each other resulted in a bit lower accuracy of the model. Every model trained nearly resulted in similar accuracy. The model trained using FastAI and transfer learning resulted in a better accuracy as compared to the other models.

Future Scope

The project is currently developed for the classification of images, and this can be extended to live video processing and real-time classification of facial expressions. With a better data set and a model with acceptable accuracy, the classification output of the project can be used to build a chat-bot whose conversations are based on the predicted emotional state of the human.

References

1. amigos-maker. (2019). *What is Flask used for?* Dev: https://dev.to/amigosmaker/what-is-flask-used-for-2do5

2. Brownlee, J. (2019). *TensorFlowTensorFlow 2 Tutorial: Get Started in Deep Learning With tf.keras.* Machine Learning Mastery: https://machinelearningmastery.com/tensorflow-tutorial-deep-learning-with-tf-keras/

3. Kulkarni, S. S., Reddy, N. P., Hariharan, S. (2009). *Facial expression (mood) recognition from facial images using committee neural networks.* BioMedical Engineering OnLine: https://biomedical-engineering-online.biomedcentral.com/articles/10.1186/1475-925X-8-16

4. Langdon, L. (2019). *Getting Started With fast.ai.* Towards Data Science: https://towardsdatascience.com/getting-started-with-fast-ai-350914ee65d2

5. Manning, E. (2018). *Are emotionally intelligent bots the future of AI?* RACONTEUR: https://www.raconteur.net/technology/artificial-intelligence/emotionally-intelligent-ai-future/

6. Ruiz, P. (2018). *Understanding and visualizing ResNets.* Towards Data Science: https://towardsdatascience.com/understanding-and-visualizing-resnets-442284831be8