Facial Emotion Recognition Using Transfer Learning and comparing the Quantized model

Nishant Chauhan
Department of Electrical Engineering
IIT Delhi
Hauz khas, Delhi
eet212377@ee.iitd.ac.in

I. Introduction

Facial expression recognition from static images is a challenging computer vision problem.Facial expression has a lot of real world application in real world like studying the human behaviour being one of themIn this project, we have used transfer learning technique in order to create a model capable of recognizing the human facial expression. Finally we use the trained model, and create a quantized model from this and compare the results. .

II. Dataset

In this problem, I have used the CK+ dataset. In 2000, the Cohn-Kanade (CK) database was released for the purpose of promoting research into automatically detecting individual facial expressions. This is one of the best and oldest dataset used for the face expression recognition problem. It has a total of 981 images and which has images for 7 different expression.

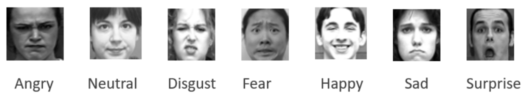


Fig. 1. Different expressions inour dataset

III. Image Processing Technique used

A. CLAHE

Histogram equalization is one of the tools we have for image pre-processing and it makes image thresholding or segmentation tasks easier.The reason we need histogram equalization is that when we collect images that are washed out or images with low contrast, we can stretch the histogram to span the entire range. This is the adaptive histogram technique but the problem with this technique is that a lot of noise is there so we use CLAHE technique. CLAHE is Contrast Limited Adaptive histogram technique. In this instead of using global contrast we use local contrast in the smaller section of image called tiles.

Identify applicable funding agency here. If none, delete this.

B. Face Crop

Now our next next image processing technique is Face crop. We first detect the face in a particular image by using a openCV and deep learning model.We will use the caffe pre-trained model for face detection.Based on the output detection we crop the face in a particular image.

C. Data Augmentation

Now since in our dataset there are very less no of images i.e. only 981 images. So we apply data augmentation technique using this we increase the no of images by rotating the original images at different angles and applying translation on the image.

IV. Transfer Learning

The reuse of a previously learned model on a new problem is known as transfer learning. It's particularly popular in deep learning right now since it can train deep neural networks with a small amount of data. This is particularly valuable in the field of data science, as most real-world situations do not require millions of labelled data points to train complicated models. The knowledge of an already trained machine learning model is transferred to a different but closely linked problem throughout transfer learning. For example, if you trained a simple classifier to predict whether an image contains a backpack, you could use the model's training knowledge to identify other objects such as sunglasses. In computer vision, neural networks typically aim to detect edges in the first layer, forms in the middle layer, and task-specific features in the latter layers. The early and central layers are employed in transfer learning, and the latter layers are only retrained. It makes use of the labelled data from the task it was trained on. In this problem we have used the 3 models namely MobleNet, Resnet50 and Vgg16 .

V. Quantization

The fundamental idea behind quantization is that if we convert the weights and inputs into integer types, we consume less memory and on certain hardware, the calculations are faster. However, there is a trade-off: with quantization, we can lose significant accuracy. We will dive into this later, but first let's see why quantization works. The idea is very simple in principle. (Not so much in practice, as we'll see later.) Suppose that you have a layer with outputs in the range of [-a, a), where a is any real number. First, we scale the output to [-128, 128), then we simply round down. That is, we use the transformation But the problem is that quantization works

$$x \mapsto \left\lfloor 128 \frac{x}{a} \right\rfloor$$

Fig. 2.

on approximations so there is a little difference in the original output and the approximate integer output. There are 2 types of quantization.

A. Quantization Aware Technique

quantize the weights during training. Here, even the gradients are calculated for the quantized weights. When applying int8 quantization, this has the best result, but it is more involved than the other option.

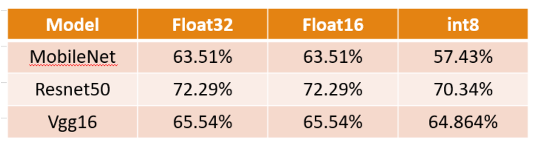
B. Post Training Quantization

Train the model using float32 weights and inputs, then quantize the weights. Its main advantage that it is simple to apply. Downside is, it can result in accuracy loss.

In this problem our focus will be on how post training quantization effects the model size effects the model size and performance

VI. Results and performance

After applying the image processing techniques we will fed the images to the model and training is performed for float32 model. After training is done for float32 we will quantize the model and we will get 2 different model. One for float16 and one for int8. Below table shows us the accuracy of different models when the weights and layers output are in float32, float16 and int8 format. Now as we can see the accuracy for



Model	Float32	Float16	int8
MobileNet	63.51%	63.51%	57.43%
Resnet50	72.29%	72.29%	70.34%
Vgg16	65.54%	65.54%	64.864%

Fig. 3.

float32 and float16 is same but there is a drop in the accuracy of int8 model which is quite an expected result as when float32 to int8 conversion is done there is a lot of approximation and there is a differenece in actual weights and approximated int8 weights. Now coming to size of different models we have a quite a great pattern. As we can see the chart for each kind of model, the size of float16 is half of the size of float32 model and size of int8 model is half the size of float16 model.

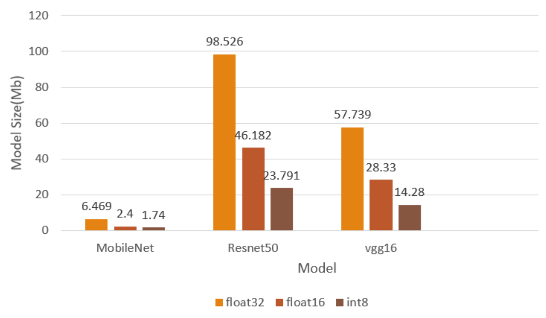


Fig. 4.

VII. Conclusion

While applying the quantization technique one thing that need to be kept in mind is that there is a trade off between the quantized model size and the accuracy. As we compressed our model to lower precision value, the approximation will happen and this approximation might results in lower accuracy as compared to float32 model.

References

[1] https://www.kaggle.com/datasets/shawon10/ckplus
[2] https://arxiv.org/abs/2104.11274
[3] https://arxiv.org/abs/1804.08348
[4] https://towardsdatascience.com/how-to-accelerate-and-compress-neural-networks-with-quantization-edfbbabb6af7
[5] https://www.tensorflow.org/lite/performance/post$trainingquantization$