CS 783 Assignment 2

Somesh Kumar Jaishwal(18111071) and Harshit Gupta(18111020)

28 February 2019

Side Details:

1. Images resized to 224x224
2. Batch size fixed to 16
3. 20% dataset for validation
4. 'Accuracy' mentioned anywhere in this report denotes 'Validation Accuracy'
5. Pretrained CNN models used are pretrained on **imagenet** dataset
6. 'Final model' mentioned anywhere in this report denotes model that consists of both coarse grain as well as fine grain classifier.

1 Methods tried

1.1 For coarse grained classification

1.1.1 Via Naive CNN

We crafted CNNs variants with 3-4 convolutional layers and experimented with different number of filters in each layer. The best accuracy we got in this method is approximately 88% via network with 3 convolutional layers of 32, 64, 64 filters each of size 3x3 with ReLU activation units, followed by a dense layer of size 1024 with softmax activation unit.

1.1.2 Via Pretrained Model

Among various pre-trained models that we tried (MobileNet(and MobileNetV2), Xception, VGG19 and VGG16), the final model VGG16 has total number of parameters less than 22 million. Also, it gave comparatively better accuracy(98-99%) for coarse grained classification. We used VGG 16 in these ways:

1. Used VGG16 convolutional base features followed by 2 dense layers(128 and 5 neurons) with 5% dropout. Though the accuracy is good, the results were not stable as the accuracy changed a lot with every epoch in range(90-99%).
2. Fine tuning last two layers of convolutional base and then applying dense layers as above. Two approaches:
   (a) Tuned last 2 layer first for coarse grained classification, thereafter used the convolutional base after freezing all layers) for fine grained classification. Accuracy varies as 92-94-97-99% over epochs.
   (b) Tuned last 2 layer first for fine grained classification, thereafter used the convolutional base after freezing all layers) for coarse grained classification. Accuracy seemed to be most stable in this approach. Accuracy varies approximately from 97.5 to 99%.

1.1.3 Via Convolutional Base of Bilinear CNN trained for Fine grained classification

This approach is described later in section 1.2.3 where we've described bilinear CNN approach for fine grain classification.

1.2 For Fine grained classification

1.2.1 Via Naive CNN

We tried crafting CNN for every subclass. Performed poor as less data for each subclass (also low diversity in examples over intra-classes).

1.2.2 Via Pretrained CNN

The approach is partly described in section 1.1.2-2 (a) and (b). In approach (a), We concatenated the coarse grained results (a probability vector) with image features from convolutional base, and feed it to a neural network with 2 layers (128 neuroned(ReLU) layer followed by 36 neuroned, softmaxed layer) with dropout rate 0.5. The accuracy seemed to be bounded by 68%. In approach (b), We got fine grained accuracy bounded by 81%. Also, Tuning layers on fine grained classification (36 classes) forces it to be more informative about our dataset, thats why, we guess, coarse classification accuracy mentioned in section 1.1.2-2 (b) seemed to be most stable.

1.2.3 Via Bilinear CNN

1. Twice VGG16 convolutional base for Bilinearity : 36 subclasses. Steps :
   (a) Freezed convolutional base, and learned the bilinear layer in 5 epochs.
   (b) Freezed bilinear layer, and tuned last 4 layer of convolutional base in 20 epochs.

   Fine grained classification accuracy was bounded by 83% (average ~80%) but the total number parameters (~28M) of final modal exceeded 22M. Coarse grain classification is done by using the same convolution base features followed by 2 dense layers(128 neuroned, ReLu-ed layer and 5 neuroned, softmaxed layer) with 5% dropout. Coarse grain classification accuracy is approximately 98%.

2. Twice MobileNetV2 convolutional base for Bilinearity: 36 subclasses. It has 88 layers (Advantage : Can remove layers as needed). Steps :
   (a) Disabled last 3 layers to reduce the total number of parameters while learing bilinearity. Original number of total parameters > 50M. Reduced number of total parameter $\simeq 12$M.
   (b) Freezed convolutional base (the 3 layer disabled version from (a)), and learned the bilinear layer in 15 epochs.
   (c) Freezed bilinear layer, and tuned last 4 layer of convolutional base (the last 4 layer after disabling last 3 layer) in 30 epochs.

   Fine grained classification accuracy is approximately 90%. Coarse grain classification is done by using the full convolutional base (disabled layers are now enabled) features followed by same set of layers as in above approach. Coarse grain classification accuracy is approximately 98%.

2 Finally chosen model(s)

Bilinear CNN using MobileNetV2 architecture as convolutional base. The last enabled layer of convolution base (after disabling layers as mentioned in section 1.2.3-2) is followed by a bilinear layer and a softmaxed layer with 36 neurons for fine grained clasification. The convolutional base (all layers, diasabled ones are now enabled) is followed by a two dense layers (128 neuroned, ReLu-ed Layer and 5 neuron, softmaxed layer) for coarse grained classification.

3 Tries to improve Accuracy:

1. We found approach described in section 1.1.2. (b) improves fine grained accuracy (improved to 81%). This approach also stabilized coarse classification accuracy.
2. Analyzed layers of pretrained convolutional base for tuning. Tried to find layers that stores less information (possibly the ones with less number and size of filters) and other than tuning last few layers, tuning these for our dataset. This sort of inclines the pretrained convolutional base towards our dataset without destroying their pre-learned more informative features of dataset (imagenet) that is superset of ours.

4 Problems Faced

1. Tuning parameter of handcrafted/naive CNNs. Also, sometimes adding an extra layer/neuron intended for improving classification actually worsened it.

2. Identifying layers in convolutional base for tuning to align the same to our dataset.

3. Tuning layers of dense network (following Convolutional Base) and augmenting layers of convolution base to restrict number of total parameters to 22M.

4. We found that training these models is computationally very expensive and given that GPUs were occupied all the time we were having hard time trying to train our models. After struggling for a long time with this, we found out about Google Collab and trained our model there.