CS783 Assignment 1

Anubhav Mittal, Tanuj

February 27th, 2019

1 Approaches Used

1.1 Coarse grained classification - I - Bag of Visual words with ORB descriptors

We first looked at ORB (we used ORB as it is freely available) to obtain local features/ feature vector and then classify the query image via TF-IDF score as discussed in class. We first extracted the visual descriptors from each image in the training set. The cluster centers were obtained via the K-Means Clustering. Various values of number of cluster centers were tried and we use k=50 .
We then represented each image by a historgram in the following way:first created a vector of k value for each image and then for each keypoints in an image, found the nearest center and increase by one its value, and finally normalised it.
We use these vectors as the embeddings of the image, and use a neural network to train a classifier for this. This model performs very poorly with train accuracy of around 30%:

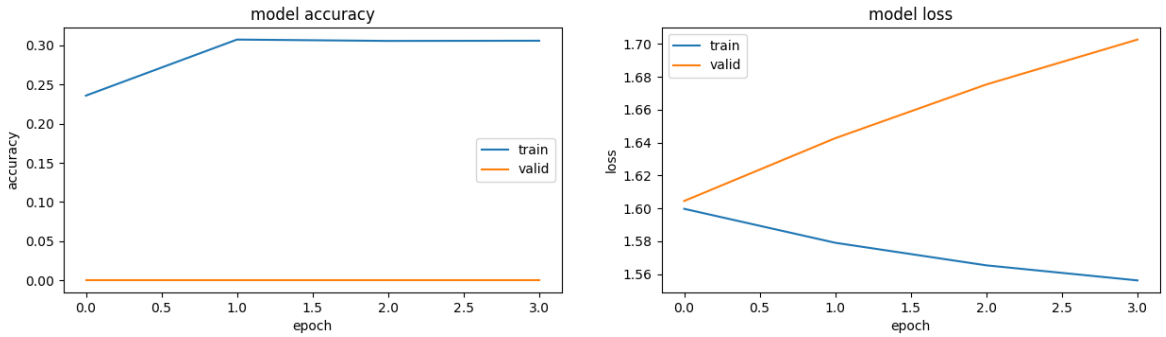

Figure 1: Accuracy and Loss vs iterations for training of the classifier layer

The code for this is present in the file "bagOfWords.py".

1.2 Coarse grained classification - II

We use ResNet18 as a feature vector extractor, and add an average 2D pooling layer and a dense fully connected layer which outputs a vector equal to no. of coarse classes=5. We leave the weights of the Resnet18 unchanged and only train the weights of the last classifier layer that we introduced. Using this method, we obtain an accuracy of 95.28%:

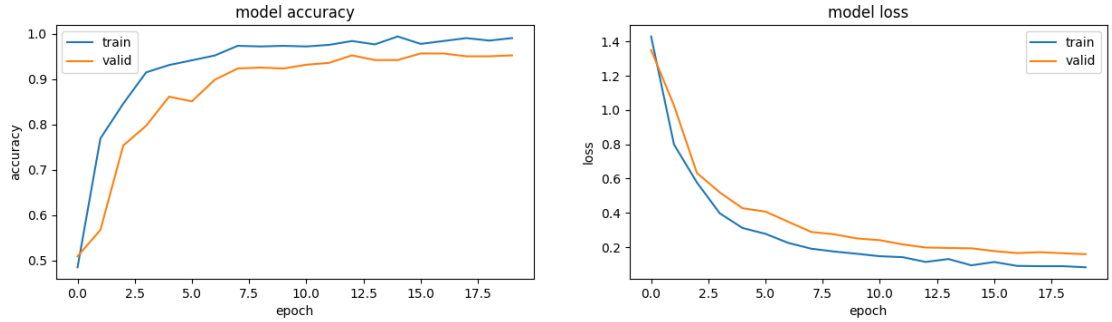

Figure 2: Accuracy and Loss vs iterations for training of the classifier layer

This is implemented in "Coarse_classification.py".

1.3 Fine Grained Classification - I

We take the approach similar to defined above, except no. of classes defined here is 36 and the ResNet18 Layers are also tuned:

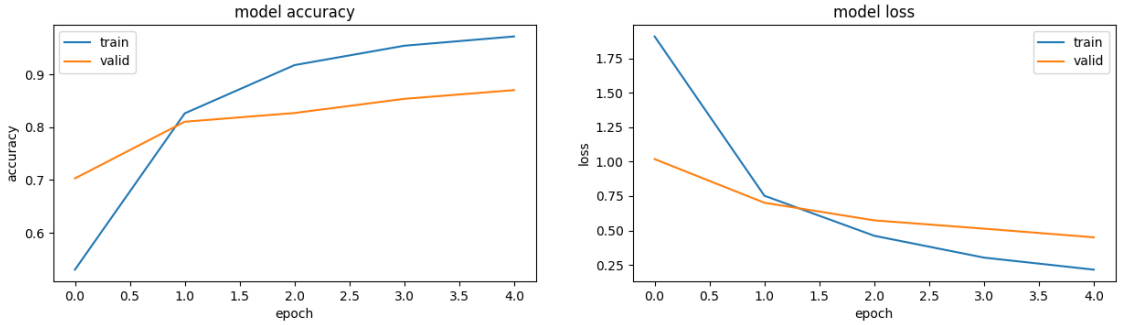

Figure 3: Accuracy and Loss vs iterations for training of the Network

We obtain an accuracy of 87.01% using this method. More importantly, we obtain a ResNet weight set fine tuned to our classes. We use this in our improvements. The code for this is present in "fine_classification.py".

1.4 Coarse grained classification - III

1.4.1 Neural Network approach

This is similar to the 1.2 approach, except we now use the feature extractor given by model learnt in 1.3 and learn a single layer network which takes the feature vector as input and outputs a vector equal to no. of coarse classes=5. The accuracy increases to 99.79%:

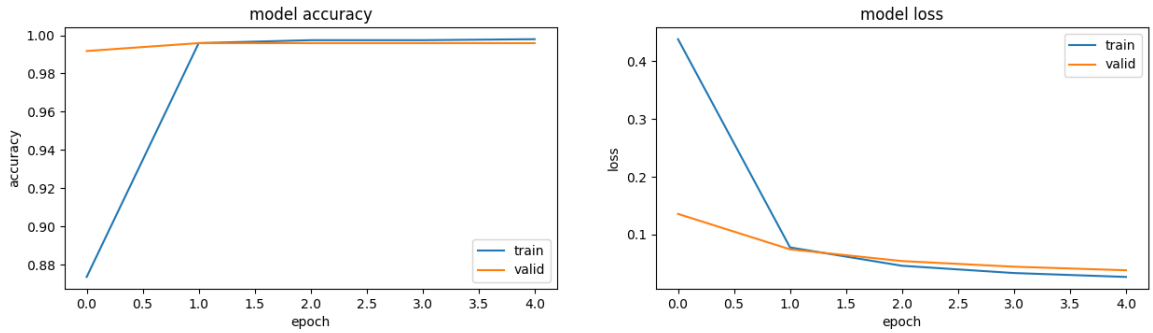

Figure 4: Accuracy and Loss vs iterations for training of the classifier

The code for this is present in "Coarse_classification_2.py". This saves the model in .hdf5 form, and I had trouble loading this in my final classifier. I tried to use pickle but did not have any success. Hence, I redid this with a slight change:

1.4.2 SVM approach

The appraoch is similar to the one used above, but instead of a Neural net layer as our classifier, we use an SVM. The accuracy comes out to be the same as above: 99.79%. This implementation is present in the file "Coarse_classification_3.py".

1.5 Fine grained classification - II

We use a two step procedure in this method. We train separate classifiers for each of the five main classes which classify the sub-classes. Then we do the following: First, using the coarse grained classifier-III, we decide which main class the image belongs to. Then, using the classifier trained for the sub-classes of that class, we obtain the sub-class for the image.

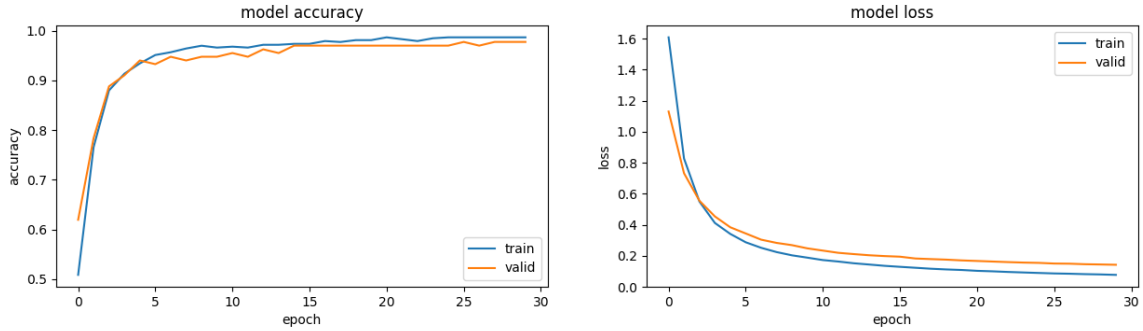

Figure 5: Accuracy and Loss vs iterations for training of the classifier for aircrafts

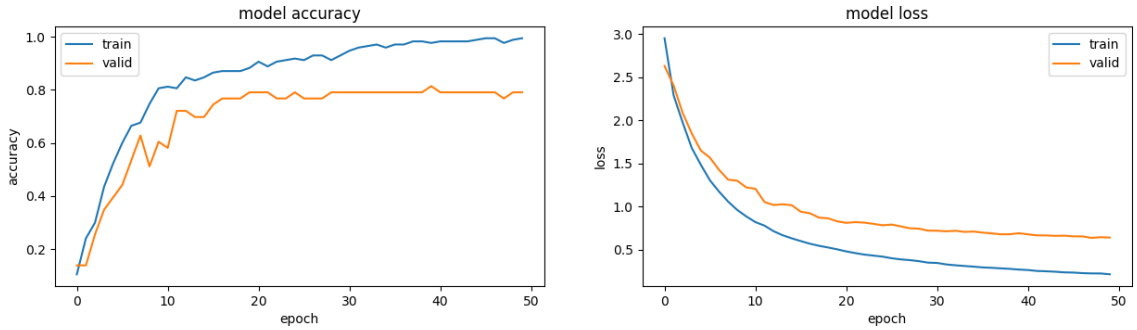

Figure 6: Accuracy and Loss vs iterations for training of the classifier for birds

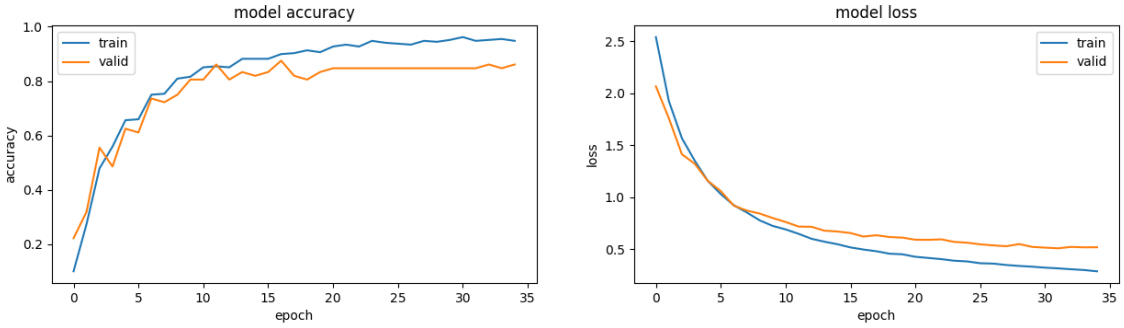

Figure 7: Accuracy and Loss vs iterations for training of the classifier for cars

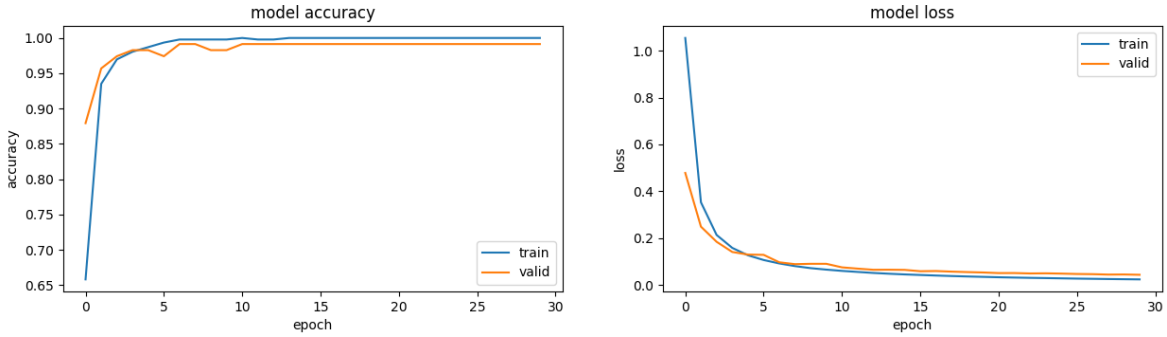

Figure 8: Accuracy and Loss vs iterations for training of the classifier for dogs

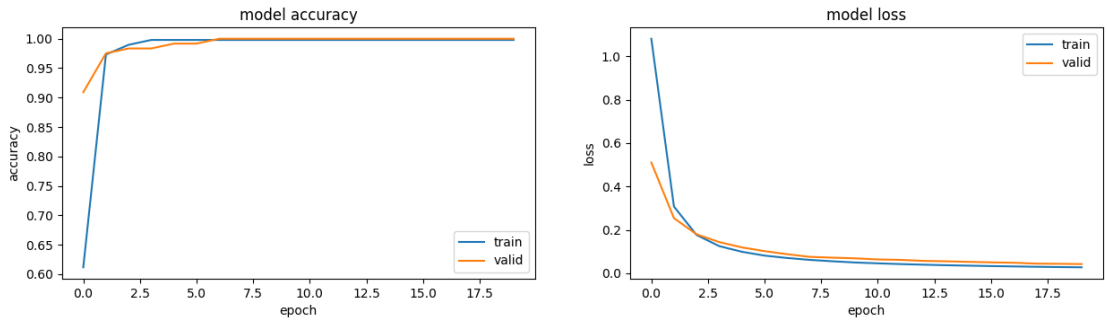

Figure 9: Accuracy and Loss vs iterations for training of the classifier for flowers

This again had the same issues as 1.3 . Hence, again I trained SVM classifier for each sub-class and proceeded.

Class	Neural Net Classifier	SVM Classifier
aircrafts	97.76	97.01
birds	79.07	72.09
cars	86.11	83.33
dogs	99.14	100
flowers	100	100

Out of the 2423 images given, the above model correctly predicted the coarse AND fine label of 2370 images.

2 Challenges Faced

The initial approach using bag of words performed very poorly with about 0.3 accuracy. The models involving the ResNet18 feature extractor worked fine, and when we finely tuned the entire network, the coarse classification accuracy reached close to perfect. After little tweaks in the pipeline for fine grain classification using a two-step classification instead of 1, increased fine grain classification to a very high accuracy as well.