Jaden He

11-775 HW 2

Mar 3rd, 2022

Pipeline

1. Extract SIFT features from 8249 .mp4 videos in data/videos using run_sift.py.
2. Train K-means for 128 and 1024 clusters respectively from extracted SIFT features using train_kmeans.py.
3. Extract Bag-of-Words representation from trained K-means model sift_128.pkl and sift_1024.pkl using run_bow.py, store them in bow_sift_128 and bow_sift_1024 respectively. 1/3 of the features were sampled.
4. Normalized Bag-of-Words representation using tools/norm.py, and store them in bow_sift_128_normed and bow_sift_1024_normed.
5. Extract CNN features from 8249 .mp4 videos in data/videos using pretrained models resnet18, resnet101, and resnet152 using run_cnn.py. Store them in cnn_resnet18, cnn_resnet101, and cnn_resnet152. All CNN features were preprocessed using transform of ImageNet. Change model selection in get_stages fuction.
6. Train the classification models on MLP from above mentioned features using run_mlp.py respectively. The MLP model are based on 4 layers, used batch normalization and ReLu activation. Kaiming normal weight and AdamW optimizer with weight_decay=0.05 were used. ReduceLROnPlateau scheduler was used, and the learning rate was set to 4e-3. num_features=128 for bow_sift_128 and bow_sift_128_normed, num_features=1024 for bow_sift_1024 and bow_sift_1024_normed. num_features=512 for cnn_resnet18,

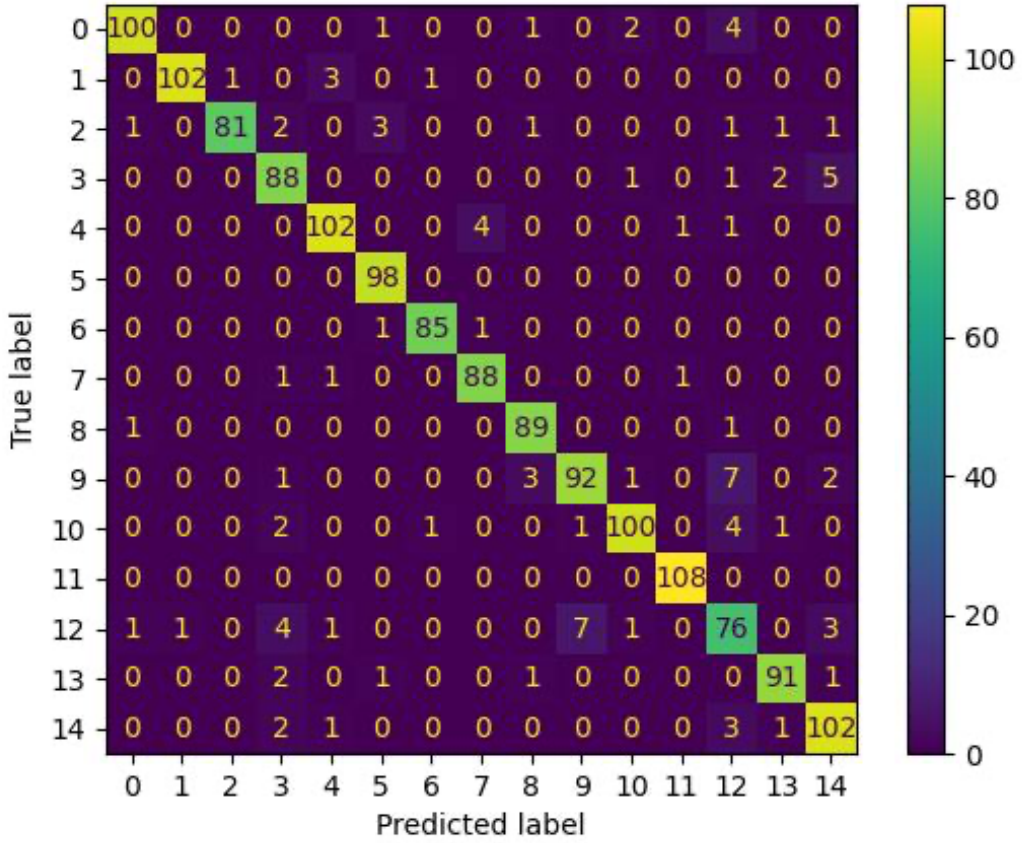
num_features=2048 for cnn_resnet101 and cnn_resnet152. Set max_epochs=150 and batch_size=1024 for them.

7. Extract CNN features using resnet152 from 29694 .mp4 videos in data/videos_p2, also store them in cnn_resnet152, and run it on MLP.
8. Evaluate the performance of all models by Top-1 accuracy metric, taking test_frac=0.2, 80% of the set to train and 20% to validate.
9. Generate confusion matrix for the models using tools/confution_matrix.py.

Table of results

Model	Features	Best Validation Accuracy	Test Accuracy
Part 1			
MLP-SIFT-BoW-128	SIFT-128	0.277	0.09821
MLP-SIFT-BoW-128_normed	SIFT-128_normed	0.275	0.07142
MLP-SIFT-BoW-1024	SIFT-1024	0.305	0.39285
MLP-SIFT-BoW-1024_normed	SIFT-1024_normed	0.303	0.36160
MLP-CNN-resnet18	CNN-resnet18-512	0.909	0.89285
MLP-CNN-resnet101	CNN-resnet18-2048	0.929	0.91517
MLP-CNN-resnet152	CNN-resnet18-2048	0.935	0.92410
Part 2			
MLP-CNN-resnet152	CNN-resnet18-2048	0.932	

Confusion matrix and best model



The model with the highest best validation top-1 accuracy is the MLP with 4 layers with batch normalization and ReLU activation, hidden layer size of 2048, Kaiming normal weight, optimizer with weight_decay=0.05, ReduceLROnPlateau scheduler, learning rate of 4e-3, max_epochs=150, and batch_size=1024, trained from CNN resnet152 features. To reproduce the results, train MLP using python code/run_mlp.py cnn --feature_dir data/cnn_resnet152 --num_features=2048 --max_epochs=300 --batch_size=1024. The process should take about 184 seconds on AWS g4dn.4xlarge instance.

Overall, this model did a pretty good job on our 15-class classification. There are barely places with more than 5 misclassifications in the confusion matrix. Only three places have 5 or more misclassified labels. This is understandable since there are videos of people singing while playing guitar. It is also possible that there are multiple people in the picture, some people are singing, and some people are playing guitar. I think in this case the model might be more likely to classify it as playing guitar, since the guitar is more recognizable than a person singing with their mouth open. In fact, in HW1, our sound model was also prone to misclassify these two classes, but it works better on this visual model. Another point is 3 (getting_a_haircut) and 14 (tickling). This may be because at least two people are involved in these two activities, and the distance between the two people is close. The body movements of people in these two activities are quite similar as well. In conclusion, it is surprised for me to see the visual models can achieve such high accuracy in video classification.

Insights and Error analysis

From the experimental results of SIFT, the performance of this hand-crated feature is indeed performed worse than neural network's features. At first, I set the number of clusters in K-means to 128. Although it achieved 0.277 on the best validation accuracy, it did not reach 0.1 on the test accuracy. I initially thought that the data of the features may fluctuate too much, so I normalized the Bag of Words, and it turns out that the normed result is worse than the original. Later, I thought that it might be difficult to classify the classes because the number of clusters was too small, so I increased the number of clusters to 1024, and the result reached 0.305 on the best validation

accuracy and close to 0.4 on the test accuracy. I tried normalization again on cluster number of 1024, and the result was still worse than the original. I think it may be because normalization narrows the difference between the features, making the results worse.

Regarding the features extracted using resnet, I spent quite a lot of time on torchvision's documentation to understand how to normalize input so that those pre-trained models could work, and figure out the different dimensions of avgpool for different versions of resnet. Overall, deeper and wider neural networks with more parameters did achieve higher accuracy on our dataset.

In part 2, I choose to use resnet152 to extract features from videos_p2. However, 4 out of 29694 videos could not be extracted. I checked what's wrong with these 4 videos, and found "NjM0NTkzODMwMTIzMDAyOTcwNg==" is a folder with 50 videos that are not in our .csv file. And "LTI3NTA5MjY3MDQwMTgwMDg5MjY=", "NzIyODk0Mjc2MTE4OTM3NTk4", "ODU0NjMxNTczMDI0MTQwMjE5MA==" three videos are probably damaged that without any contents. So I decided to delete these 4 videos in the prediction .csv, and made a new one named "test_for_students_modified.csv", and I performed the prediction task on this new .csv file. I included these 4 videos in videos_lack folder under labels_p2 folder.

Running time

1. Extracting SIFT features from labels/train_val.csv took 3614 seconds, from labels/ test_for_students.csv took 383 seconds on c5.4xlarge instance.

2. Running Kmeans of 128 cluster took 662 second, Kmeans of 1024 clusters took 4629 seconds on g4dn.4xlarge instance.
3. Extracting BoW features from both sift_128.pkl and sift_1024.pkl took about 3 - 4 minutes respectively on g4dn.4xlarge instance.
4. Running MLP with bow_sift_128 and bow_sift_1024, with or without norm, took about 95 seconds and 180 seconds on g4dn.4xlarge instance.
5. Extracting resnet18 features from labels/train_val.csv took 6753 seconds, from labels/ test_for_students.csv took 232 seconds on g4dn.4xlarge instance.
6. Running MLP with cnn_resnet18 took 136 seconds on g4dn.4xlarge instance.
7. Extracting resnet101 features from labels/train_val.csv took 2210 seconds, from labels/ test_for_students.csv took 231 seconds on g4dn.4xlarge instance.
8. Running MLP with cnn_resnet101 took 184 seconds on g4dn.4xlarge instance.
9. Extracting resnet152 features from labels/train_val.csv took 3068 seconds, from labels/ test_for_students.csv took 313 seconds on g4dn.4xlarge instance.
10. Running MLP with cnn_resnet152 took 131 seconds on g4dn.4xlarge instance.
11. Extracting resnet152 features from labels_p2/ test_for_students.csv took 12231 seconds, and running MLP for data_p2.csv took 142 seconds on g4dn.4xlarge instance.

AWS credits

I spent $2.14 on EFS, $11.06 on t2.2xlarge and c5.4xlarge instances, and $22.102 on g4dn.4xlarge instance, in total $35.302. I have all my $50 AWS credits from HW1, so I have $64.698 on my account.