HW2 Report

11775-ISR: Large Scale Multimedia Analysis ISR Section

10/10/2022
Gyuseok Lee

1. Introduction

This homework is designed for which method extracts feature well. There are three methods in this homework: SIFT, CNN, CNN3D. After extracting the video features through each method, its features inputs into MLP model for training, and then I can evaluate the MLP model performance for video classification. Consequentially, each performance is 0.3258(SIFT), 0.9278(CNN), and 0.94385(CNN3D) in Kaggle. In the remaining sections, I would like to explain dataset, model, experiment for each method, and conclusion.

2. Dataset

I split the dataset into train and validation set. The size of each dataset is 6,000 (Train) and 1,500 (Validation). After the training and validation, the model predicts the test dataset. The size of test dataset is 750.

3. Model

Although each extraction method is different, the MLP models for classification are very similar. The model architecture is like below figure. Model is two layers MLP model because I think its embedding is already fine-grained feature. So, I decided to classify the video via shallow network. Specifically, I used the pretrained *convnext(base)* for CNN and *R2Plus1D18* for CNN3d. The input channel of each model (SIFT, CNN, CNN3D) is 128, 1024, 512. Note that the first layer is followed by BatchNorm1D, GELU, and Dropout. Plus, I used AdamW optimizer which is one of the best optimizers in these days and CosineAnnealingWarmRestarts as a scheduler. The initial learning rate is 0.01.

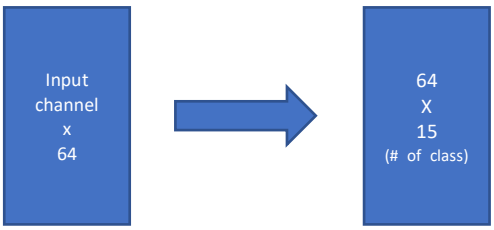

Figure1. Model Architecture

4. Experiment

1) SIFT

There are several steps for using SIFT and testing the model. Frist, extract the SIFT feature from the video file. Each video frame would be with a dimension of 32 x 128 (where 32 and 128 mean the location and dimension, respectively). Second, cluster each location using the K-means and proceed with the bag of words. At this point, there are 128 cluster we set up and 32 x 128 frames would be 1 x 128 vectors (W). So, np.sum(W) would be 32 because we predict the cluster for each frame (1 x. 128). After that, I stacked all the frames in the video and perform maximize for axis = 0 to extract the fine-grained video feature for video 128 Dimension. Finally, this video feature (128 dimensions 1D vector) would be used for training and validating. I attached the Confusion matrix and the table for several important information (i.e., Accuracy, Precision, Recall and F1-Score).
The top-1 accuracy is 0.336 (validation). The hardest class is 2 and the easiest class is 11. The training time which applied early stopping is 144.02 seconds.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
0	18	6	8	13	2	7	4	9	8	1	5	5	4	9	9
1	3	31	5	10	4	5	5	3	4	5	5	11	1	5	10
2	6	9	11	9	2	7	3	3	10	5	6	3	1	11	5
3	3	11	6	35	1	0	3	2	1	2	7	6	1	7	12
4	9	6	3	5	27	3	16	17	7	1	3	5	4	0	2
5	4	4	7	2	2	36	9	10	2	5	5	4	1	3	4
6	6	2	1	8	6	5	28	11	6	1	1	5	5	0	2
7	7	4	3	5	9	5	11	32	7	2	1	2	0	1	2
8	3	1	1	1	8	3	5	5	50	5	3	4	1	0	1
9	10	3	1	4	4	10	7	3	10	31	9	2	2	3	7
10	1	4	4	5	3	2	3	0	13	7	44	6	5	11	1
11	4	10	3	5	2	1	4	2	2	1	2	62	3	1	6
12	7	12	3	5	8	2	4	5	10	6	4	7	11	4	6
13	2	8	2	8	2	2	0	1	1	1	7	2	2	42	16
14	6	11	2	12	1	3	4	1	3	5	1	2	5	7	46

Table1. Confusion Matrix for SIFT

	TP	FP	TN	FN	Acurracy	Precision	Recall	F1-Score
0	18	71	90	1321	0.012000	0.202247	0.013443	0.025210
1	31	91	76	1302	0.020667	0.254098	0.023256	0.042612
2	11	49	80	1360	0.007333	0.183333	0.008023	0.015374
3	35	92	62	1311	0.023333	0.275591	0.026003	0.047522
4	27	54	81	1338	0.018000	0.333333	0.019780	0.037344
5	36	55	62	1347	0.024000	0.395604	0.026030	0.048847
6	28	78	59	1335	0.018667	0.264151	0.020543	0.038121
7	32	72	59	1337	0.021333	0.307692	0.023375	0.043449
8	50	84	41	1325	0.033333	0.373134	0.036364	0.066269
9	31	47	75	1347	0.020667	0.397436	0.022496	0.042582
10	44	59	65	1332	0.029333	0.427184	0.031977	0.059500
11	62	64	46	1328	0.041333	0.492063	0.044604	0.081794
12	11	35	83	1371	0.007333	0.239130	0.007959	0.015406
13	42	62	54	1342	0.028000	0.403846	0.030347	0.056452
14	46	83	63	1308	0.030667	0.356589	0.033973	0.062036

Table2. Descriptive Statics Table for SIFT

2) CNN

As I already mentioned, I used the *convnext(base)* which was pretrained with ImageNet dataset. So, the embedding size is 1024. At the bottom of the line is that the top-1 accuracy is 0.94 for validation set and the training time for convergence is 144 seconds. The hardest class is 12 and the easiest class is 11. I attached the confusion matrix and descriptive statistics table below.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
0	104	0	1	1	0	0	0	0	0	0	0	0	0	0	2
1	0	106	0	0	0	0	0	0	0	0	1	0	0	0	0
2	1	0	86	0	0	2	0	0	0	0	0	0	0	0	2
3	2	0	1	85	0	0	0	0	0	2	0	0	2	2	3
4	0	2	0	0	101	0	0	5	0	0	0	0	0	0	0
5	0	0	2	0	0	96	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	1	82	1	0	0	0	0	2	1	0
7	0	0	0	0	2	0	1	86	0	0	0	0	1	1	0
8	0	0	0	0	0	0	0	0	90	0	0	0	1	0	0
9	0	0	0	0	0	0	0	0	2	94	0	0	9	0	1
10	0	0	0	0	0	0	0	0	0	1	102	0	5	1	0
11	0	0	0	0	0	0	0	0	0	0	0	108	0	0	0
12	2	1	0	3	0	0	0	0	0	6	3	0	78	0	1
13	0	0	0	1	0	0	0	0	0	0	1	0	1	93	0
14	2	0	0	2	1	1	0	0	0	0	0	0	1	2	100

Table3. Confusion Matrix for CNN

	TP	FP	TN	FN	Acurracy	Precision	Recall	F1-Score
0	104	7	4	1385	0.069333	0.936937	0.069846	0.130000
1	106	3	1	1390	0.070667	0.972477	0.070856	0.132087
2	86	4	5	1405	0.057333	0.955556	0.057679	0.108792
3	85	7	12	1396	0.056667	0.923913	0.057394	0.108074
4	101	3	7	1389	0.067333	0.971154	0.067785	0.126725
5	96	4	2	1398	0.064000	0.960000	0.064257	0.120452
6	82	1	5	1412	0.054667	0.987952	0.054886	0.103995
7	86	6	5	1403	0.057333	0.934783	0.057757	0.108792
8	90	2	1	1407	0.060000	0.978261	0.060120	0.113279
9	94	9	12	1385	0.062667	0.912621	0.063556	0.118837
10	102	5	7	1386	0.068000	0.953271	0.068548	0.127900
11	108	0	0	1392	0.072000	1.000000	0.072000	0.134328
12	78	22	16	1384	0.052000	0.780000	0.053352	0.099872
13	93	7	3	1397	0.062000	0.930000	0.062416	0.116981
14	100	9	9	1382	0.066667	0.917431	0.067476	0.125707

Table4. Descriptive Statics Table for CNN

3) CNN3D

I used the pretrained *R2Plus1D18* for CNN3d which has the 512 dimensions for embedding. The top-1 accuracy is 0.982 and the training time for convergence is 174.6. The hardest class is 8 and the easiest class is 0,10,11,14. I attached the confusion matrix and descriptive statistics table below.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
0	108	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	107	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	89	0	0	0	0	0	1	0	0	0	0	0	1
3	0	0	0	97	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	107	0	0	1	0	0	0	0	0	0	0
5	0	0	3	0	0	94	0	0	0	0	0	0	0	1	0
6	0	0	0	0	0	0	86	1	0	0	0	0	0	0	0
7	0	0	0	0	0	1	0	90	0	0	0	0	0	0	0
8	0	0	1	0	0	0	1	0	85	0	0	0	4	0	0
9	0	0	0	0	0	0	0	0	2	102	0	0	2	0	0
10	0	0	0	0	0	0	0	0	0	0	108	0	1	0	0
11	0	0	0	0	0	0	0	0	0	0	0	108	0	0	0
12	1	0	1	0	0	0	0	0	0	1	1	1	89	0	0
13	0	0	0	0	0	0	0	0	0	1	0	0	0	95	0
14	0	0	0	0	0	0	0	1	0	0	0	0	0	0	108

Table5. Confusion Matrix for CNN3D

	TP	FP	TN	FN	Acurracy	Precision	Recall	F1-Score
0	108	1	0	1391	0.072000	0.990826	0.072048	0.134328
1	107	0	0	1393	0.071333	1.000000	0.071333	0.133167
2	89	5	2	1404	0.059333	0.946809	0.059612	0.112161
3	97	0	0	1403	0.064667	1.000000	0.064667	0.121478
4	107	0	1	1392	0.071333	1.000000	0.071381	0.133250
5	94	1	4	1401	0.062667	0.989474	0.062876	0.118239
6	86	1	1	1412	0.057333	0.988506	0.057410	0.108517
7	90	3	1	1406	0.060000	0.967742	0.060160	0.113279
8	85	3	6	1406	0.056667	0.965909	0.057009	0.107663
9	102	2	4	1392	0.068000	0.980769	0.068273	0.127660
10	108	1	1	1390	0.072000	0.990826	0.072096	0.134412
11	108	1	0	1391	0.072000	0.990826	0.072048	0.134328
12	89	7	5	1399	0.059333	0.927083	0.059812	0.112374
13	95	1	1	1403	0.063333	0.989583	0.063418	0.119197
14	108	1	1	1390	0.072000	0.990826	0.072096	0.134412

Table6. Descriptive Statics Table for CNN3D

5. Conclusion

In summary, I extract the video feature by using each method like SIFT, CNN, and CNN3D. Even though there are several and various methods for extracting, embedding representation made by CNN3D is the most fine-grained and its performance is the best as 0.98 for validation set. Through this homework, I have learned how to extract the embedding and realized that the fine-grained dataset is the most important thing even though there are big deep learning models and fancy techniques. In the Future, I would like to study the effective and efficient feature extracting techniques for my research.