Homework #1: Musical Instrument Recognition

1. Algorithm Description

1.1 Feature Extraction

Before extracting the features, I randomly chose one sound sample for each of ten class, and listen to the sounds and checked all spectrograms.(feature_check.ipynb) With the naked eye, the default parameter setting of STFT(Window size=1024, Hop size=512, FFT size=1024) was enough to distinguish all of the instruments, so I kept using that parameter as starting point of my feature extraction. In total, I used MFCC, delta MFCC, double delta MFCC, and Spectral Centroid. For extracting all feature, I only used librosa.features library.

MFCC, delta MFCC, double delta MFCC

Also, I found that the pitch of each sound with same instrument class in the training set was different, so I used MFCC which contains the spectral information except pitch, rather than directly using Mel-Spectrogram. I tried to add the Mel-Spectrogram in addition to MFCC, but didn't find any significant accuracy improvement compared to increased number of features(>128). Moreover, for some cases, the accuracy was decreased when I added Mel-Spectrogram.

Therefore, instead of not using Mel-Spectrogram in my algorithm, I used bigger MFCC bin size 30, than default settings, 20. However, when I checked the MFCC spectrogram for each class, I thought that only using MFCC is not enough to distinguish all class, so I added delta MFCC and double delta MFCC.

Spectral Centroid

Generally, timbre is as a result of the different distribution of each harmonic component, so I added spectral centroid which can explain the mean of the amplitude distribution of harmonic component.

Other Features

I tried to add some temporal feature such as tempogram and onset envelope, but there's no significant accuracy improvement compared to increased number of feature(>400). Moreover, for some cases, the accuracy was decreased when I added these features.

1.2 Feature Summary

I used mean, max, min, and codebook for feature summary. All feature were summarized and concatenated in one axis so that total training set size was (3*Feature number + K) X Number of sound file.

Mean

Mean was the default feature summary method in our homework, and it is fairly reasonable strategy to compress information over time into one value for each bin. However, I found that that there was no sound signal after 3 sec, while I was checking the spectrogram of each sound. Since the no signal after 3 sec, affects the mean value of frames, I determined not to use the entire 4 sec for each sound data. Also, when I looked the spectrogram, for most of sounds, the pattern of spectrogram was equal for 1 sec or 2 sec of sound, therefore, I determined to use only 1 sec for each sound sample for faster feature extraction and summary.

Max, Min

It is hard to obtain temporal information from feature such as ADSR in only 1 sec for each sound data. Also, taking mean of each time frame eliminates any temporal information such as attack. Therefore, I used max and min of each time frame, in addition to mean, to capture any abrupt change of each features over time. In practice, I found 4% accuracy increased after added max and min feature summary.

Codebook

With feature extraction and summary described in previous section, the algorithm didn't classify string vs. vocal. In fact, the spectral feature of the two classes are similar with the naked eye so that mean, max, min feature summary was not efficient to capture the differences in their feature. Therefore, I used another feature summary strategy, codebook.

Codebook is data compression method of K-means clustering. To get a codebook feature, first, for each time frame of feature of STFT was mapped into one of value between one to K, based on K-means clustering. After that, codebook feature with the length of K-dimentional vector was computed by counting(histogram) the number of each one to K value.

1.3 Train and Test

Classifier

For this assignment, I used Multi-Layer Perceptron model for classify the instrument, which is implemented as MLPClassifier in the sklearn.neural_network library, because of better performance of classification than other models. For detail, see *2. 1.*

Validation and Test

For hyperparameter search, validation was conducted using validation set. Validation was proceeded with changing L2 regularizer alpha, initial learning rate and the activation function. After finding appropriate hyperparameter via validation, validation and training set was merged as training set. After training, test set accuracy was calculated. For detail, see *2. 2.*

2. Experiments and Results

2.1 Classifier

For choosing a best classifier, the accuracy of SVM, KNN, and MLP on validation set were compared (**Table 1**). For SVM, different kernel and L2 regularizer alpha were used for comparing. For KNN, different algorithm were used for comparing. For MLP, different L2 regularizer alpha were used for comparing. Among different hyperparameter and different classifier, MLP had the best performance on validation set, with 77.5%(alpha = 0.005, other settings were default), therefore, MLP was also used for validation and test.

2.2 Validation and Test

After best classifier was chosen, more detailed validation was proceeded with changing initial learning rate and the activation function of MLP Classifier (**Table 2**). There were 18 conditions for initial learning rate, and 4 conditions for activation function. Among of different settings, MLP model with initial learning rate 0.1 and sigmoid activation function had the best accuracy on validation set and was chosen for final testing. After the final model was trained on merged training set, the model got 89% accuracy on the test set (**Table 3**).

SVM

alpha	accuracy(%)
0.001	73.5
0.00025	73
0.0005	**74**
0.001	74
0.0025	75
0.005	76.5
0.01	76
0.025	74.5
0.05	74.5
0.1	74.5
0.25	75.5
0.5	76
1	73
2.5	69.5
5	57.5
10	31
25	24
50	10

MLP

alpha	accuracy(%)
0.001	76
0.00025	76.5
0.0005	**77.5**
0.001	76
0.0025	74
0.005	73
0.01	70.5
0.025	75
0.05	76
0.1	72
0.25	72.5
0.5	67.5
1	69.5
2.5	69
5	76.5
10	71
25	45.5
50	21.5

SVM

kernel	accuracy(%)
linear	73.5
poly	31.5
rbf	**67.5**
sigmoid	66

KNN

algorithm	accuracy(%)
auto	67
ball tree	67
kd tree	67
brute force	67

Table 1 Validation accuracy comparison on different hyper parameters of SVM, KNN, MLP

MLP

activation function	accuracy(%)
identity	71
logistic	**77.5**
tanh	68.5
relu	72

lr	accuracy(%)
0.001	72.5
0.00025	71.5
0.0005	71.5
0.001	70.5
0.0025	70.5
0.005	71

lr	accuracy(%)
0.01	72.5
0.025	72.5
0.05	75
0.1	**76**
0.25	75.5
0.5	73.5
1	68.5
2.5	69
5	64.5
10	69.5
25	59.5
50	52.5

Table 2 Validation accuracy comparison for hyper parameter search of MLP

		PREDICTED									
		Bas	Bra	Flu	Gui	Key	Mal	Org	Ree	Str	Voc
TRUE	Bass	20	0	0	0	0	0	0	0	0	0
	Brass	0	17	1	0	0	0	0	2	0	0
	Flute	0	0	17	0	0	0	0	0	3	0
	Guitar	2	0	0	16	2	0	0	0	0	0
	Keyboard	2	0	0	0	18	0	0	0	0	0
	Mallet	0	0	0	0	0	20	0	0	0	0
	Organ	1	1	0	0	0	0	15	0	3	0
	Reed	0	0	1	0	0	0	0	17	1	1
	String	0	0	0	0	0	0	0	0	18	2
	Vocal	0	0	0	0	0	0	0	0	0	20

Table 3 Confusion matrix for each class on test set, accuracy = 89%

3. Discussion

3.1 Accuracy Improvement after using training and validation set as training set

After finding the best parameter via validation, when I combined the training and validation set as my final training set, 11.5% of accuracy was increased. For this, one of possible scenario is training set size was too small so that the classifiers couldn't learn effectively. To test this hypothesis, future work can be done by collecting more training data, or doing k-fold cross validation.

3.2 Decreased accuracy after addition of some features

As mentioned in *1.1*, Accuracy was decreased after some features are added. Also, even codebook feature summary didn't increase the accuracy in the training set. However, when I compared test accuracy with/without codebook feature for checking this issue, the accuracy with codebook feature was 3% bigger than without case, suggesting that the small dataset size was the problem, not the feature.

After these two discussion, I got to know that the dataset size is important factor for learning. Also, due to small data size, it seems deep learning is not appropriate for this homework. To test this, another future work can be done by learning simple CNN model with mel spectrogram, and check the accuracy with/without data augmentation.

4. Conclusion

I used MFCC, delta MFCC, double delta MFCC, Spectral Centroid as my feature. After that, I used mean, max, min, and also codebook to summarize the feature. After parameter search, I merged the training set and validation set as training set, and after training, I got 89% accuracy on test set. Also, during discussion, I could know that the dataset size was really important factor for machine learning.

5. References

Juhan Nam, 2020, Musical Applications of Machine Learning, Lecture Note 1-4

Librosa version 0.8.0., Feature Extraction, https://librosa.org/doc/latest/feature.html

Scikit Learn version 0.23.2, 3.1. Cross-validation: evaluating estimator performance, https://scikit-learn.org/stable/modules/cross_validation.html#computing-cross-validated-metrics