Use rectified signals. (rectify_emg_moving_average(X,20))
Use cost sensitive learning(1:10) for binary classification(0:others)
Drop some files out for test. No shuffle and split the rest data as 80% for training and 20% for validation.
Model consists of rnn and conv1d neural network.

Drop Files[6,30,31,32,33,34,35]:

Class 0 : others

Train (acc 0.989)

	Predicted 0	Predicted others
Actual 0	5789	69
Actual others	0	562

Valid (acc 0.960)

	Predicted 0	Predicted others
Actual 0	1482	62
Actual others	5	148

Test (acc 0.918)

	Predicted 0	Predicted others
Actual 0	285	41
Actual others	2	199

Class 1 : 2 : 6

Train (acc 0.962)

	Predicted 1	Predicted 2	Predicted 6
Actual 1	111	1	8
Actual 2	6	267	5
Actual 6	1	0	163

Valid (acc 0.736)

	Predicted 1	Predicted 2	Predicted 6
Actual 1	10	13	10
Actual 2	1	71	5
Actual 6	6	5	31

Test (acc 0.781)

	Predicted 1	Predicted 2	Predicted 6
Actual 1	34	8	10
Actual 2	10	45	2
Actual 6	14	0	78

Class 2 : 6

Train (acc 0.984)

	Predicted 2	Predicted 6
Actual 2	271	7
Actual 6	0	164

Valid (acc 0.899)

	Predicted 2	Predicted 6
Actual 2	71	6
Actual 6	6	36

Test (acc 0.959)

	Predicted 2	Predicted 6
Actual 2	52	5
Actual 6	1	91

Class 1 : 6

Train (acc 0.883)

	Predicted 1	Predicted 6
Actual 1	90	30
Actual 6	3	161

Valid (acc 0.693)

	Predicted 1	Predicted 6
Actual 1	19	14
Actual 6	9	33

Test (acc 0.902)

	Predicted 1	Predicted 6
Actual 1	45	7
Actual 6	7	85

Class 1 : 2

Train (acc 0.904)

	Predicted 1	Predicted 2
Actual 1	116	4
Actual 2	34	244

Valid (acc 0.863)

	Predicted 1	Predicted 2
Actual 1	24	9
Actual 2	6	71

Test (acc 0.724)

	Predicted 1	Predicted 2
Actual 1	50	2
Actual 2	28	29

Drop Files[7,30,31,32,33,34,35]:

Class 0 : others

Train (acc 0.986)

	Predicted 0	Predicted others
Actual 0	5815	83
Actual others	2	465

Valid (acc 0.954)

	Predicted 0	Predicted others
Actual 0	1492	62
Actual others	14	115

Test (acc 0.959)

	Predicted 0	Predicted others
Actual 0	265	11
Actual others	13	307

Class 1 : 2 : 6

Train (acc 1.000)

	Predicted 1	Predicted 2	Predicted 6
Actual 1	23	16	22
Actual 2	0	247	8
Actual 6	0	0	151

Valid (acc 0.926)

	Predicted 1	Predicted 2	Predicted 6
Actual 1	1	9	8
Actual 2	0	69	3
Actual 6	2	5	32

Test (acc 0.655)

	Predicted 1	Predicted 2	Predicted 6
Actual 1	3	48	75
Actual 2	0	66	19
Actual 6	0	0	108

Class 2 : 6

Train (acc 0.997)

	Predicted 2	Predicted 6
Actual 2	254	1
Actual 6	0	151

Valid (acc 0.891)

	Predicted 2	Predicted 6
Actual 2	69	3
Actual 6	9	30

Test (acc 0.906)

	Predicted 2	Predicted 6
Actual 2	68	17
Actual 6	1	107

Class 1 : 6

Train (acc 1.000)

	Predicted 1	Predicted 6
Actual 1	61	0
Actual 6	0	151

Valid (acc 0.807)

	Predicted 1	Predicted 6
Actual 1	15	3
Actual 6	8	31

Test (acc 0.756)

	Predicted 1	Predicted 6
Actual 1	72	54
Actual 6	3	105

Class 1 : 2

Train (acc 1.000)

	Predicted 1	Predicted 2
Actual 1	61	0
Actual 2	0	255

Valid (acc 0.900)

	Predicted 1	Predicted 2
Actual 1	11	7
Actual 2	2	70

Test (acc 0.668)

	Predicted 1	Predicted 2
Actual 1	91	35
Actual 2	35	50

Drop Files[5,30,31,32,33,34,35]:

Class 0 : others

Train (acc 0.977)

	Predicted 0	Predicted others
Actual 0	5724	146
Actual others	0	629

Valid (acc 0.929)

	Predicted 0	Predicted others
Actual 0	1433	114
Actual others	7	164

Test (acc 0.918)

	Predicted 0	Predicted others
Actual 0	283	28
Actual others	7	109

Class 1 : 2 : 6

Train (acc 0.890)

	Predicted 1	Predicted 2	Predicted 6
Actual 1	94	32	25
Actual 2	6	305	5
Actual 6	1	0	161

Valid (acc 0.788)

	Predicted 1	Predicted 2	Predicted 6
Actual 1	10	18	13
Actual 2	1	86	0
Actual 6	2	2	38

Test (acc 0.370)

	Predicted 1	Predicted 2	Predicted 6
Actual 1	0	13	0
Actual 2	2	6	1
Actual 6	22	35	37

Class 2 : 6

Train (acc 0.997)

	Predicted 2	Predicted 6
Actual 2	315	1
Actual 6	0	162

Valid (acc 0.976)

	Predicted 2	Predicted 6
Actual 2	86	1
Actual 6	2	40

Test (acc 0.553)

	Predicted 2	Predicted 6
Actual 2	8	1
Actual 6	45	49

Class 1 : 6

Train (acc 0.920)

	Predicted 1	Predicted 6
Actual 1	141	10
Actual 6	15	147

Valid (acc 0.843)

	Predicted 1	Predicted 6
Actual 1	31	10
Actual 6	3	39

Test (acc 0.327)

	Predicted 1	Predicted 6
Actual 1	13	0
Actual 6	72	22

Class 1 : 2

Train (acc 0.937)

	Predicted 1	Predicted 2
Actual 1	148	3
Actual 2	26	290

Valid (acc 0.882)

	Predicted 1	Predicted 2
Actual 1	30	11
Actual 2	4	83

Test (acc 0.318)

	Predicted 1	Predicted 2
Actual 1	0	13
Actual 2	2	7

Model:

```
kernel_size=3
reg=regularizers.l2(1e-4)
drop_rate = 0.
kernel_initializer = 'glorot_normal'
mo = 0.8
st = 1
axis = 2
model = keras.models.Sequential()
model.add(layers.InputLayer(input_shape=X[:,:,:].shape[1:]))
model.add(layers.Bidirectional(layers.LSTM(32,return_sequences=True,
recurrent_regularizer=reg)))
model.add(layers.Conv1D(filters=32, kernel_size=kernel_size,
                        strides=st,
                        padding='same',
                        kernel_regularizer=reg,
                       ))
model.add(layers.BatchNormalization(momentum=mo))
model.add(layers.LeakyReLU(0.1))
model.add(layers.MaxPooling1D(2))
model.add(layers.Dropout(drop_rate))
model.add(layers.Conv1D(filters=16, kernel_size=kernel_size,
                        strides=st,
                        padding='same',
                        kernel_regularizer=reg,
                       ))
model.add(layers.BatchNormalization(momentum=mo))
model.add(layers.LeakyReLU(0.1))
model.add(layers.MaxPooling1D(2))
model.add(layers.Dropout(drop_rate))
model.add(layers.Conv1D(filters=8, kernel_size=kernel_size,
                        strides=st,
                        padding='same',
                        kernel_regularizer=reg,
                       ))
model.add(layers.BatchNormalization(momentum=mo))
```

```
model.add(layers.LeakyReLU(0.1))
model.add(layers.MaxPooling1D(2))
model.add(layers.Dropout(drop_rate))
model.add(layers.Conv1D(filters=4, kernel_size=kernel_size,
                        strides=st,
                        padding='same',
                        kernel_regularizer=reg,
                       ))
model.add(layers.BatchNormalization(momentum=mo))
model.add(layers.LeakyReLU(0.1))
model.add(layers.MaxPooling1D(2))
model.add(layers.GlobalAveragePooling1D())
model.add(layers.Dropout(drop_rate))
model.add(layers.Dense(2,activation='softmax',kernel_regularizer=reg))
```