Window size:1024
Stride: 512
Raw data with detrend(lambda:300)
Use all channels
No scale on raw signal. Shuffle and split data to 20% for test. Shuffle and split the rest to 80% for training and 20% for validation.

```
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=1,padding='same',kernel_regularizer=reg))
model.add(layers.BatchNormalization(momentum=0.8))
model.add(layers.ELU())
model.add(layers.AveragePooling1D(2))
model.add(layers.Dropout(0.5))
model.add(layers.Conv1D(filters=16,
kernel_size=kernel_size,strides=1,padding='same',kernel_regularizer=reg))
model.add(layers.BatchNormalization(momentum=0.8))
model.add(layers.ELU())
model.add(layers.AveragePooling1D(2))
model.add(layers.Dropout(0.5))
model.add(layers.Conv1D(filters=8,
kernel_size=kernel_size,strides=1,padding='same',kernel_regularizer=reg))
model.add(layers.BatchNormalization(momentum=0.8))
model.add(layers.ELU())
model.add(layers.AveragePooling1D(2))
model.add(layers.Dropout(0.5))
model.add(layers.Conv1D(filters=4,
kernel_size=kernel_size,strides=1,padding='same',kernel_regularizer=reg))
model.add(layers.BatchNormalization(momentum=0.8))
model.add(layers.ELU())
model.add(layers.GlobalAveragePooling1D())
model.add(layers.Dropout(0.5))
model.add(layers.Dense(3,activation='softmax',kernel_regularizer=reg))
```

```
kernel_size=7
reg=regularizers.l2(0)
```

Train (acc 1.000)

	Predicted 1	Predicted 2	Predicted 6	acc
Actual 1	117	0	0	1.000
Actual 2	0	324	0	1.000
Actual 6	0	0	183	1.000

Validation (acc 0.885)

	Predicted 1	Predicted 2	Predicted 6	acc
Actual 1	38	11	2	0.745
Actual 2	5	85	0	0.944
Actual 6	5	1	62	0.911

Test 20% (acc 0.856)

	Predicted 1	Predicted 2	Predicted 6	acc
Actual 1	31	9	9	0.632
Actual 2	5	92	2	0.929
Actual 6	4	1	56	0.918

kernel_size=7

Reg = regularizers.l2(1e-4)

Train (acc 1.000)

	Predicted 1	Predicted 2	Predicted 6	acc
Actual 1	117	0	0	1.000
Actual 2	0	324	0	1.000
Actual 6	0	0	183	1.000

Validation (acc 0.889)

	Predicted 1	Predicted 2	Predicted 6	acc
Actual 1	41	6	4	0.803
Actual 2	5	85	0	0.944
Actual 6	7	1	60	0.882

Test 20% (acc 0.866)

	Predicted 1	Predicted 2	Predicted 6	acc
Actual 1	32	10	7	0.653
Actual 2	6	93	0	0.939
Actual 6	4	1	56	0.918

kernel_size=9

reg=regularizers.l2(0)

Train (acc 1.000)

	Predicted 1	Predicted 2	Predicted 6	acc
Actual 1	117	0	0	1.000
Actual 2	0	324	0	1.000
Actual 6	0	0	183	1.000

Validation (acc 0.870)

	Predicted 1	Predicted 2	Predicted 6	acc
Actual 1	35	11	5	0.686
Actual 2	5	84	1	0.933
Actual 6	4	1	63	0.926

Test 20% (acc 0.889)

	Predicted 1	Predicted 2	Predicted 6	acc
Actual 1	34	7	8	0.693
Actual 2	2	95	2	0.959
Actual 6	3	1	57	0.934

kernel_size=9
reg=regularizers.l2(1e-4)

Train (acc 1.000)

	Predicted 1	Predicted 2	Predicted 6	acc
Actual 1	117	0	0	1.000
Actual 2	0	324	0	1.000
Actual 6	0	0	183	1.000

Validation (acc 0.894)

	Predicted 1	Predicted 2	Predicted 6	acc
Actual 1	39	9	3	0.764
Actual 2	4	85	1	0.944
Actual 6	4	1	63	0.926

Test 20% (acc 0.880)

	Predicted 1	Predicted 2	Predicted 6	acc
Actual 1	34	9	6	0.693
Actual 2	5	91	3	0.919
Actual 6	1	1	59	0.967

kernel_size=9
reg=regularizers.l2(1e-4)
class weights: 1:5, 2:1, 6:1

Train (acc 0.996)

	Predicted 1	Predicted 2	Predicted 6	acc
Actual 1	117	0	0	1.000
Actual 2	0	324	0	1.000
Actual 6	1	0	182	0.994

Validation (acc 0.904)

	Predicted 1	Predicted 2	Predicted 6	acc
Actual 1	41	7	1	0.803
Actual 2	6	83	1	0.922
Actual 6	5	0	63	0.926

Test 20% (acc 0.894)

	Predicted 1	Predicted 2	Predicted 6	acc
Actual 1	41	5	3	0.836
Actual 2	5	92	2	0.929
Actual 6	4	3	54	0.885

kernel_size=9
reg=regularizers.l2(1e-4)
class weights: 1:10, 2:1, 6:1

Train (acc 0.982)

	Predicted 1	Predicted 2	Predicted 6	acc
Actual 1	117	0	0	1.000
Actual 2	0	324	0	1.000
Actual 6	1	10	172	0.939

Validation (acc 0.861)

	Predicted 1	Predicted 2	Predicted 6	acc
Actual 1	43	5	3	0.843
Actual 2	3	85	2	0.944
Actual 6	4	5	59	0.867

Test 20% (acc 0.885)

	Predicted 1	Predicted 2	Predicted 6	acc
Actual 1	44	2	3	0.897
Actual 2	5	92	2	0.929
Actual 6	6	5	50	0.819

```
kernel_size=9
reg=regularizers.l2(1e-4)
cost_matrix = ([[0,    3.,    1.5],
                [1,    0,    1],
                [1.,    1., 0]])
```

Train (acc 1.000)

	Predicted 1	Predicted 2	Predicted 6	acc
Actual 1	117	0	0	1.000
Actual 2	0	324	0	1.000
Actual 6	0	0	183	1.000

Validation (acc 0.904)

	Predicted 1	Predicted 2	Predicted 6	acc
Actual 1	44	4	3	0.862
Actual 2	7	81	2	0.900
Actual 6	4	0	64	0.941

Test 20% (acc 0.875)

	Predicted 1	Predicted 2	Predicted 6	acc
Actual 1	39	3	7	0.795
Actual 2	7	89	3	0.898
Actual 6	5	1	55	0.901

```
input_ = layers.Input(shape=X[:,:,:].shape[1:],name='input')
lstm = layers.Bidirectional(layers.LSTM(32,return_sequences=True,
                                        name = 'lstm',
                                       recurrent_regularizer=reg))(input_)
cnn11 = layers.Conv1D(filters=32,
kernel_size=kernel_size1,strides=1,padding='same',kernel_regularizer=reg)(lstm)
cnn11 = layers.BatchNormalization(momentum=0.8,name='BN_11')(cnn11)
cnn11 = layers.ELU(name='Act_11')(cnn11)
cnn11 = layers.AveragePooling1D(2,name='pooling_11')(cnn11)
cnn11 = layers.Dropout(0.5,name='dropout_11')(cnn11)

cnn12 = layers.Conv1D(filters=32,
kernel_size=kernel_size2,strides=1,padding='same',kernel_regularizer=reg)(lstm)
cnn12 = layers.BatchNormalization(momentum=0.8,name='BN_12')(cnn12)
cnn12 = layers.ELU(name='Act_12')(cnn12)
cnn12 = layers.AveragePooling1D(2,name='pooling_12')(cnn12)
cnn12 = layers.Dropout(0.5,name='dropout_12')(cnn12)

cnn21 = layers.Conv1D(filters=16,
```

```
kernel_size=kernel_size1,strides=1,padding='same',kernel_regularizer=reg)(cnn11)
cnn21 = layers.BatchNormalization(momentum=0.8,name='BN_21')(cnn21)
cnn21 = layers.ELU(name='Act_21')(cnn21)
cnn21 = layers.AveragePooling1D(2,name='pooling_21')(cnn21)
cnn21 = layers.Dropout(0.5,name='dropout_21')(cnn21)

cnn22 = layers.Conv1D(filters=16,
kernel_size=kernel_size2,strides=1,padding='same',kernel_regularizer=reg)(cnn12)
cnn22 = layers.BatchNormalization(momentum=0.8,name='BN_22')(cnn22)
cnn22 = layers.ELU(name='Act_22')(cnn22)
cnn22 = layers.AveragePooling1D(2,name='pooling_22')(cnn22)
cnn22 = layers.Dropout(0.5,name='dropout_22')(cnn22)

cnn31 = layers.Conv1D(filters=8,
kernel_size=kernel_size1,strides=1,padding='same',kernel_regularizer=reg)(cnn21)
cnn31 = layers.BatchNormalization(momentum=0.8,name='BN_31')(cnn31)
cnn31 = layers.ELU(name='Act_31')(cnn31)
cnn31 = layers.AveragePooling1D(2,name='pooling_31')(cnn31)
cnn31 = layers.Dropout(0.5,name='dropout_31')(cnn31)

cnn32 = layers.Conv1D(filters=8,
kernel_size=kernel_size2,strides=1,padding='same',kernel_regularizer=reg)(cnn22)
cnn32 = layers.BatchNormalization(momentum=0.8,name='BN_32')(cnn32)
cnn32 = layers.ELU(name='Act_32')(cnn32)
cnn32 = layers.AveragePooling1D(2,name='pooling_32')(cnn32)
cnn32 = layers.Dropout(0.5,name='dropout_32')(cnn32)

cnn41 = layers.Conv1D(filters=4,
kernel_size=kernel_size1,strides=1,padding='same',kernel_regularizer=reg)(cnn31)
cnn41 = layers.BatchNormalization(momentum=0.8,name='BN_41')(cnn41)
cnn41 = layers.ELU(name='Act_41')(cnn41)

cnn42 = layers.Conv1D(filters=4,
kernel_size=kernel_size1,strides=1,padding='same',kernel_regularizer=reg)(cnn32)
cnn42 = layers.BatchNormalization(momentum=0.8,name='BN_42')(cnn42)
cnn42 = layers.ELU(name='Act_42')(cnn42)

cnn4 = layers.concatenate([cnn41,cnn42],axis=2)

cnn4 = layers.GlobalAveragePooling1D(name='global_average')(cnn4)
cnn4 = layers.Dropout(0.5,name='dropout_4')(cnn4)

output = layers.Dense(3,activation='softmax',kernel_regularizer=reg,name='output')(cnn4)
model = Model(input_, output)
```

kernel_size1=11
kernel_size2=7
reg=regularizers.l2(1e-4)

Train (acc 1.000)

	Predicted 1	Predicted 2	Predicted 6	acc
Actual 1	117	0	0	1.000
Actual 2	0	324	0	1.000
Actual 6	0	0	183	1.000

Validation (acc 0.875)

	Predicted 1	Predicted 2	Predicted 6	acc
Actual 1	35	8	6	0.714
Actual 2	7	82	1	0.911
Actual 6	4	0	64	0.941

Test 20% (acc 0.870)

	Predicted 1	Predicted 2	Predicted 6	acc
Actual 1	35	8	6	0.714
Actual 2	6	91	2	0.919
Actual 6	3	2	56	0.918

kernel_size1=11
kernel_size2=7
reg=regularizers.l2(1e-4)
class weights: 1:5, 2:1, 6:1

Train (acc 0.996)

	Predicted 1	Predicted 2	Predicted 6	acc
Actual 1	117	0	0	1.000
Actual 2	0	324	0	1.000
Actual 6	0	2	181	0.989

Validation (acc 0.856)

	Predicted 1	Predicted 2	Predicted 6	acc
Actual 1	41	7	3	0.803
Actual 2	5	83	2	0.922
Actual 6	9	4	55	0.808

Test 20% (acc 0.889)

	Predicted 1	Predicted 2	Predicted 6	acc
Actual 1	43	3	3	0.877
Actual 2	6	91	2	0.919
Actual 6	5	4	52	0.852

kernel_size1=11
kernel_size2=7
reg=regularizers.l2(1e-4)
class weights: 1:10, 2:1, 6:1

Train (acc 0.988)

	Predicted 1	Predicted 2	Predicted 6	acc
Actual 1	117	0	0	1.000
Actual 2	0	323	1	0.996
Actual 6	0	6	177	0.967

Validation (acc 0.870)

	Predicted 1	Predicted 2	Predicted 6	acc
Actual 1	43	6	2	0.843
Actual 2	8	81	1	0.900
Actual 6	5	5	58	0.852

Test 20% (acc 0.870)

	Predicted 1	Predicted 2	Predicted 6	acc
Actual 1	41	4	4	0.836
Actual 2	8	88	3	0.888
Actual 6	3	5	53	0.868

```
def inception_module(input_,filters,reg):

    x1 = layers.Conv1D(filters=filters[0], kernel_size=3, strides=1,
activation=None,padding='same',kernel_regularizer=reg)(input_)

    x2 = layers.Conv1D(filters=filters[1], kernel_size=7, strides=1, activation=None,
padding='same',kernel_regularizer=reg)(input_)

    x3 = layers.Conv1D(filters=filters[2], kernel_size=11, strides=1, activation=None,
padding='same',kernel_regularizer=reg)(input_)

    x4 = layers.Conv1D(filters=filters[3], kernel_size=15, strides=1, activation=None,
padding='same',kernel_regularizer=reg)(input_)

    merge = layers.concatenate([x1,x2,x3,x4],axis=2)

    return merge

reg=regularizers.l2(1e-3)
input_ = layers.Input(shape=X[:,:,:].shape[1:],name='input')
lstm = layers.Bidirectional(layers.LSTM(32,return_sequences=True,
                                        name = 'lstm',
                                       recurrent_regularizer=reg))(input_)

cnn1 = inception_module(lstm,[32,16,8,8],reg)
cnn1 = layers.BatchNormalization(momentum=0.8,name='BN_1')(cnn1)
cnn1 = layers.ELU(name='Act_1')(cnn1)
cnn1 = layers.AveragePooling1D(2,name='pooling_1')(cnn1)
cnn1 = layers.Dropout(0.5,name='dropout_1')(cnn1)

cnn2 = inception_module(cnn1,[16,8,4,4],reg)
cnn2 = layers.BatchNormalization(momentum=0.8,name='BN_2')(cnn2)
cnn2 = layers.ELU(name='Act_2')(cnn2)
cnn2 = layers.AveragePooling1D(2,name='pooling_2')(cnn2)
cnn2 = layers.Dropout(0.5,name='dropout_2')(cnn2)

cnn3 = inception_module(cnn2,[8,4,2,2],reg)
cnn3 = layers.BatchNormalization(momentum=0.8,name='BN_3')(cnn3)
cnn3 = layers.ELU(name='Act_3')(cnn3)
cnn3 = layers.AveragePooling1D(2,name='pooling_3')(cnn3)
cnn3 = layers.Dropout(0.5,name='dropout_3')(cnn3)

cnn4 = inception_module(cnn3,[4,2,1,1],reg)
cnn4 = layers.BatchNormalization(momentum=0.8,name='BN_4')(cnn4)
```

```
cnn4 = layers.ELU(name='Act_4')(cnn4)
cnn4 = layers.GlobalAveragePooling1D(name='global_average')(cnn4)
cnn4 = layers.Dropout(0.5,name='dropout_4')(cnn4)

output = layers.Dense(3,activation='softmax',kernel_regularizer=reg,name='output')(cnn4)
model = Model(input_, output)
```

Train (acc 1.000)

	Predicted 1	Predicted 2	Predicted 6	acc
Actual 1	117	0	0	1.000
Actual 2	0	324	0	1.000
Actual 6	0	0	183	1.000

Validation (acc 0.904)

	Predicted 1	Predicted 2	Predicted 6	acc
Actual 1	41	6	4	0.803
Actual 2	5	82	3	0.911
Actual 6	2	0	66	0.970

Test 20% (acc 0.842)

	Predicted 1	Predicted 2	Predicted 6	acc
Actual 1	30	10	9	0.625
Actual 2	3	89	7	0.898
Actual 6	3	1	57	0.934

class weights: 1:5, 2:1, 6:1

Train (acc 0.998)

	Predicted 1	Predicted 2	Predicted 6	acc
Actual 1	117	0	0	1.000
Actual 2	0	323	1	0.996
Actual 6	0	0	183	1.000

Validation (acc 0.827)

	Predicted 1	Predicted 2	Predicted 6	acc
Actual 1	36	9	6	0.705
Actual 2	10	76	4	0.844
Actual 6	5	2	61	0.897

Test 20% (acc 0.827)

	Predicted 1	Predicted 2	Predicted 6	acc
Actual 1	37	6	6	0.755
Actual 2	14	82	3	0.828
Actual 6	3	4	54	0.885