信号中有无效值，已用均值填充

最初窗口长度为260，步长30，尺度260，这样产生的数据量太大而且效果不好。在小波变换后的数据中可以看到低频很大一部分没有提供明显信息。Cnn第一层先用maxpooling会好一点，但总体还是很差。

后改为256，64，64，效果稍好，但是generalization不够好。Dropout试过0.2和0.5，0.2效果最好。将小波变换后的数据归一化的效果也不如先将emg信号归一化。

Morl小波基效果不如mexh。

最后先将emg信号归一化，小波基选为mexh，参数调为256，32，32，去掉第一层maxpooling后效果最好。

使用standard scaler来预处理emg信号，效果也很好。

## 13.07:

用其他病人的信号来测试cnn model效果不太好。对整个emg信号做standard scale效果不太好，对每个窗口做standard scale效果也不太好，在输入层和第一层之间加batch normalization效果也不行。

提取时域特征，用xgboost训练集效果很好，但generalization不够好，svm效果更差。

## 14.07：

突然发现cnn对于其他病人数据集的预测准确率只有百分之六十左右，但是xgboost有时候可以到百分之八十。

导师还有一个任务是feature engineering，所以接下来还是要多试试特征的提取。

## 15.07：

将所有非0类数据归为一类后，对xgboost做二分类测试，emg数据没有做scale。

没有预留不用的病人，将所有病人数据混合随机打散，留百分之二十做test set，剩下的百分之二十做valid，百分之八十做training。训练集准确率很高，能达到百分之九十九以上，valid集和测试集效果也可以，基本上94%到95%。用正常状态的两个数据集（label都为0）做测试能到100%。

对cnn做相同二分类测试，整个数据集做cwt太大，只选前两万个打散后训练效果不错，但是用正常状态的数据集准确率只能到百分之七十多，而且速度非常慢。测试的时候整个数据集已经做过standard scale，但是忘了所以又加了一个batch normalization层。

Cnn去掉bn层把relu改为elu，valid集和test集表现不如之前，只有93%左右，但是在正常状态的数据集中测试效果更好，有的集准确率有88%左右，但有个别数据集准确率很差。

在25000-35000数据间准确率只有72%。

## 16.07

重对SVM测试，这次修改了参数C，将C调大效果好很多，但C过大会导致过拟合。

新添加了Autogressive特征，xgboost效果更好了一点。WAMP的threshold也需要调，20似乎不错，第二天再试试AR和WAMP。

## 17.07

前一天有错误，分训练集的时候没有设定随机种子，效果有偏差。之前用的数据都是standard scale过的，基本分布都在十以内，所以WAMP超过十根本没有作用，调小以后效果更差。用没有scale过的数据效果更好了，而且用WAMP后效果有提升，而且threshold设为30比25和40都好。

但是没有scale后泛化效果变差，在没有接触过的数据集上表现不如scale后。

新添加了EMG Histogram特征，scale后该特征作用不大，但是不做scale效果提升明显。

HIST特征参数bins选9，range在（-70，70）间效果比较好。

测试了SVM，需要先将feature scale，要不然输入分布太大影响结果。XGBoost scale后泛化不如之前。

## 20.07

窗口长度200效果不好，300在测试集和valid集上表现不错，但是在其他正常的数据集和余下的2000个数据上效果不如之前。

用时域特征测试了ANN，测试集和验证集效果不错，但是在其他正常的数据集表现很差。

XGBoost lambda设为15泛化效果有所提升。Depth设为4也提升了一点泛化效果。

## 21.07

在全部数据集和全部病人但非全部数据集上分别做了测试，记录了结果。

## 07.08

新添加了MDF特征，表现有所提升。试了DNN，数据需提前标准化，否则分布太广，训练效率不高。结果与xgboost相近。试了RNN效果不太好，而且训练速度太慢。DNN激活函数elu效果比较好，selu不如elu。

## 08.08

之前没注意到有几个文件的数据无效值太多：

* G04\_FoG\_trial\_1\_emg.csv
* G04\_FoG\_trial\_2\_emg.csv
* 正常/P623\_M050\_2\_A\_Walking\_trial\_2\_emg.csv
* 正常/P623\_M050\_2\_A\_Walking\_trial\_4\_emg.csv
* 正常/P623\_M050\_2\_A\_Walking\_trial\_6\_emg.csv
* 正常/P623\_M050\_A\_Walking\_trial\_4\_emg.csv
* 正常/P623\_Msham\_A\_Walking\_trial\_6\_emg.csv

剔除这几个文件后效果有所提升。

测试了单独特征的效果，有些特征有重复：

IEMG与MAV

SSI，VAR与RMS

## 09.08

将无效值用均值替换在连续缺省的情况下会形成一条直线，尝试将无效值丢弃。

新加了mDWT和MNF特征，mDWT影响效果，MNF有所提升。

尝试剔除特征，剔除mDWT后提升最多

## 10.08

试了randomized search cv，xgboost参数：

Max depth：8

Learning rate：0.3

Subsample：0.9

Reg lambda：1

试了DWT将输出直接展开成数列当作输入，泛化效果不好，而且数据太多，PCA后也没有所提升。

新添加从cwt提取的四个特征Mean coefficient，Minimum coefficient，Mean scale，Median scale

## 11.08

Cwt新提取的四个特征提升效果不明显，几乎一样，有升有降。试了加滤波，泛化效果变差，有噪声可能对于泛化效果提升有帮助。

Cwt试了两种scale长度，32和64，32效果最好。

Dnn不用dropout三个隐藏层加selu多分类效果比xgboost稍好，valid和test集能达到89%。

尝试提取数据时只使用两个标签都一样的数据，多分类效果有所提升。

## 02.09

G04\_FoG\_trial\_1\_emg.csv的频谱不正常，全频段均匀分布，应该抛弃看看。

低频信号特别明显可能信号有问题

## 03.09

5：75 Hz明显

3， 4，7，8，13，14，15，16：50 Hz明显

19，20 channel 3 感觉异常，频率在低频太集中。

kernel\_size=11

reg=regularizers.l2(1e-4)

drop\_rate = 0.3

kernel\_initializer = 'glorot\_normal'

model = keras.models.Sequential()

model.add(layers.InputLayer(input\_shape=X[:,:,:].shape[1:]))

model.add(layers.Bidirectional(layers.LSTM(32,return\_sequences=True,

#kernel\_regularizer=regularizers.l2(reg),

recurrent\_regularizer=reg)))

model.add(layers.Conv1D(filters=32, kernel\_size=kernel\_size,strides=2,

padding='same',

kernel\_regularizer=reg,

kernel\_initializer=kernel\_initializer))

model.add(layers.BatchNormalization(momentum=0.8))

model.add(layers.ELU())

model.add(layers.Dropout(drop\_rate))

model.add(layers.Conv1D(filters=16, kernel\_size=kernel\_size,strides=2,

padding='same',

kernel\_regularizer=reg,

kernel\_initializer=kernel\_initializer))

model.add(layers.BatchNormalization(momentum=0.8))

model.add(layers.ELU())

model.add(layers.Dropout(drop\_rate))

model.add(layers.Conv1D(filters=8, kernel\_size=kernel\_size,strides=2,

padding='same',

kernel\_regularizer=reg,

kernel\_initializer=kernel\_initializer))

model.add(layers.BatchNormalization(momentum=0.8))

model.add(layers.ELU())

model.add(layers.Dropout(drop\_rate))

model.add(layers.Conv1D(filters=4, kernel\_size=kernel\_size,strides=2,

padding='same',

kernel\_regularizer=reg,

kernel\_initializer=kernel\_initializer))

model.add(layers.BatchNormalization(momentum=0.8))

model.add(layers.ELU())

model.add(layers.GlobalAveragePooling1D())

model.add(layers.Dropout(drop\_rate))

model.add(layers.Dense(3,activation='softmax',kernel\_regularizer=reg))

width = 1024

stride = 512

no detrend

generate\_window\_slide\_data\_NA\_remove

no file df.columns[5], df.columns[17]