Gravitational Wave Detection

CMPE 257 Project
September 9, 2021

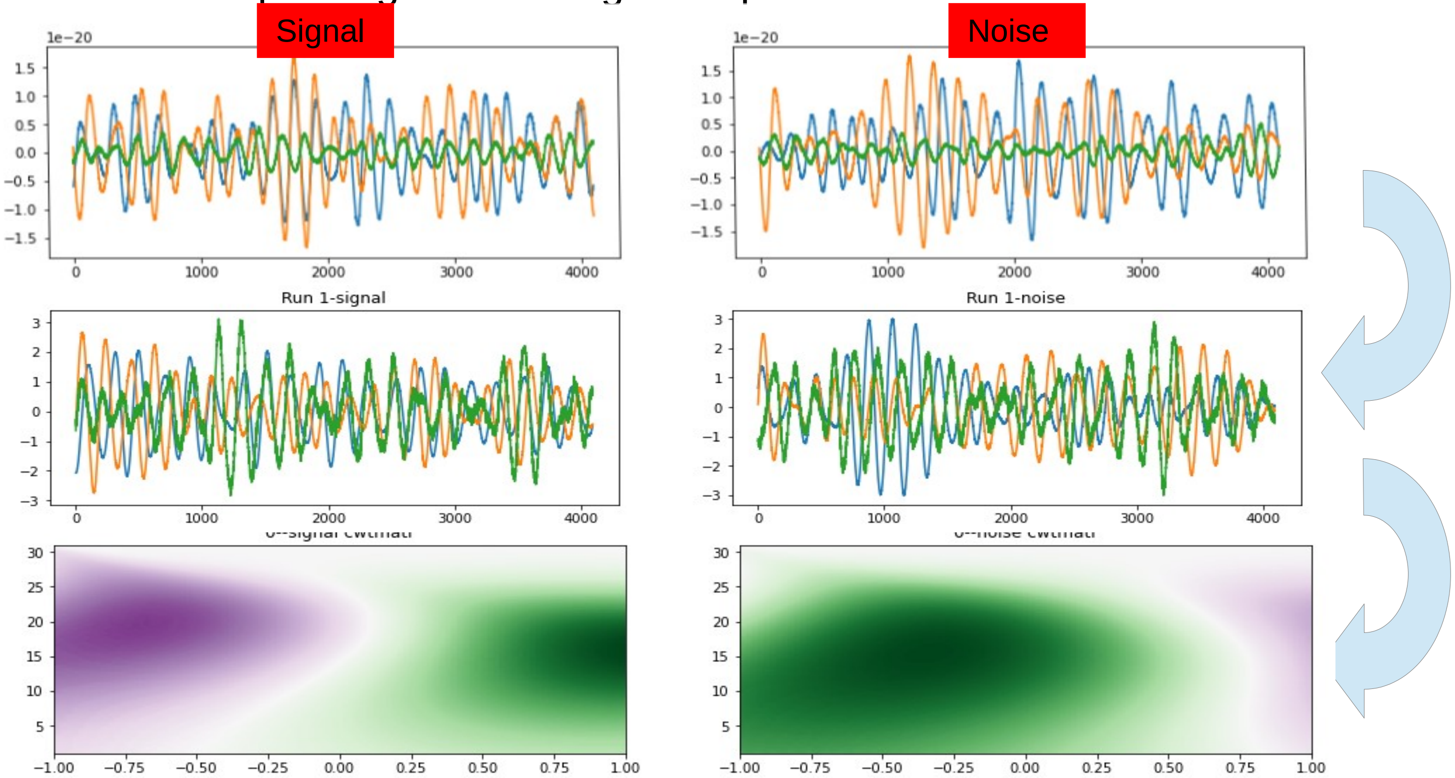
Tom Casaletto
Anthony Fisher
Phil Shirts
Joel Wiser

Outline

- Problem Statement
- CWT/CNN Approach
- CQT/CNN Approach
- Transfer Learning
- Next Steps
- Additional Material

Continuous Wavelet Transform

- Transform detector time series into spectrograms (images)
- Train CNN with spectrogram training examples

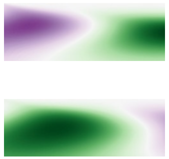


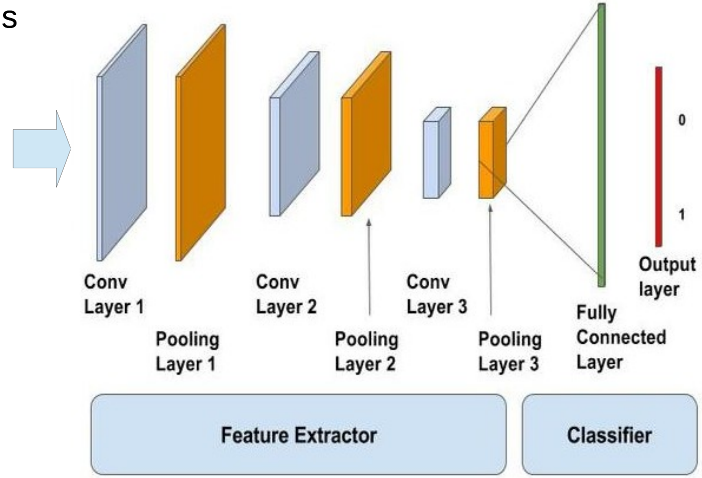
Normalize signals
(z transform)

Perform CWT transform
Apply bandpass filter
(20-500 Hz)

Convolutional Neural Net

Labeled Spectrograms





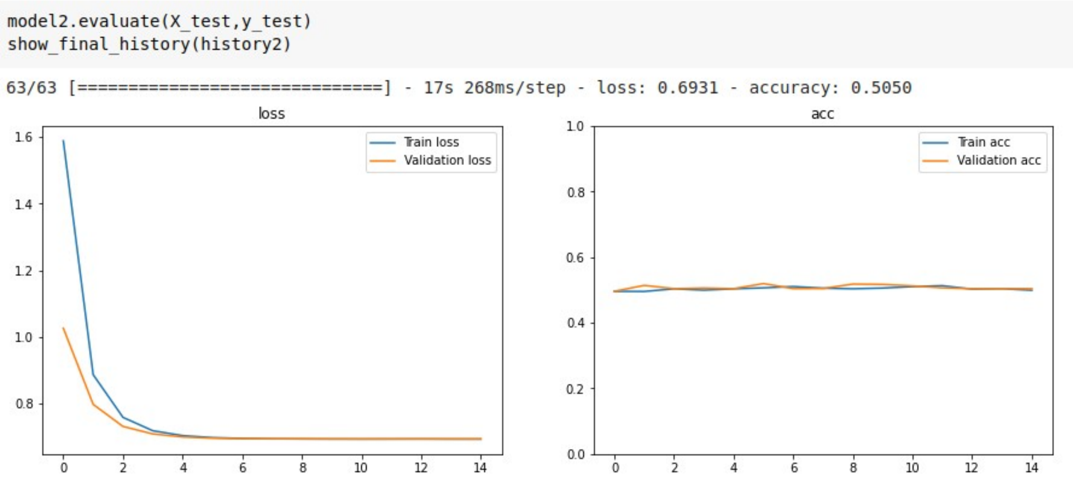
```
model = models.Sequential()
model.add(layers.Conv2D(32, (10, 10), activation='relu', input_shape=(300,97,1)))
model.add(layers.MaxPooling2D((6, 6)))
model.add(layers.Conv2D(64, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Conv2D(64, (3, 3), activation='relu'))
model.add(layers.Flatten())
model.add(layers.Dense(64, activation='relu'))
model.add(layers.Dense(2))
model.summary()

model.compile(optimizer='adam',
              loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),
              metrics=['accuracy'])
model.fit(X_train,y_train,epochs=10)
```

https://www.analyticsvidhya.com/blog/2021/08/beginners-guide-to-convolutional-neural-network-with-implementation-in-python/s

CWT/CNN Results

10000 examples, 60/20/20 split

```
Model: "sequential_13"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_16 (Conv2D)           (None, 291, 88, 32)       3232
_________________________________________________________________
max_pooling2d_11 (MaxPooling (None, 48, 14, 32)        0
_________________________________________________________________
conv2d_17 (Conv2D)           (None, 46, 12, 64)        18496
_________________________________________________________________
max_pooling2d_12 (MaxPooling (None, 23, 6, 64)         0
_________________________________________________________________
conv2d_18 (Conv2D)           (None, 21, 4, 64)         36928
_________________________________________________________________
flatten_8 (Flatten)          (None, 5376)              0
_________________________________________________________________
dense_16 (Dense)             (None, 64)                344128
_________________________________________________________________
dense_17 (Dense)             (None, 2)                 130
=================================================================
Total params: 402,914
Trainable params: 402,914
Non-trainable params: 0
_________________________________________________________________
```



```
model2.evaluate(X_test,y_test)
show_final_history(history2)
```

```
63/63 [==============================] - 17s 268ms/step - loss: 0.6931 - accuracy: 0.5050
```

```
Epoch 1/10
188/188 [==============================] - 210s 1s/step - loss: 0.7200 - accuracy: 0.5025 - val_loss: 0.6932 - val_accuracy: 0.5040
Epoch 2/10
188/188 [==============================] - 199s 1s/step - loss: 0.6959 - accuracy: 0.5040 - val_loss: 0.6923 - val_accuracy: 0.5100
Epoch 3/10
188/188 [==============================] - 199s 1s/step - loss: 0.6940 - accuracy: 0.5002 - val_loss: 0.6931 - val_accuracy: 0.5040
Epoch 4/10
188/188 [==============================] - 204s 1s/step - loss: 0.6932 - accuracy: 0.5035 - val_loss: 0.6931 - val_accuracy: 0.5040
Epoch 5/10
188/188 [==============================] - 200s 1s/step - loss: 0.6932 - accuracy: 0.5042 - val_loss: 0.6931 - val_accuracy: 0.5030
Epoch 6/10
188/188 [==============================] - 200s 1s/step - loss: 0.6933 - accuracy: 0.4997 - val_loss: 0.6932 - val_accuracy: 0.5030
Epoch 7/10
188/188 [==============================] - 200s 1s/step - loss: 0.6932 - accuracy: 0.4997 - val_loss: 0.6931 - val_accuracy: 0.5025
Epoch 8/10
188/188 [==============================] - 200s 1s/step - loss: 0.6932 - accuracy: 0.5005 - val_loss: 0.6931 - val_accuracy: 0.5035
Epoch 9/10
188/188 [==============================] - 200s 1s/step - loss: 0.6936 - accuracy: 0.5010 - val_loss: 0.6933 - val_accuracy: 0.4960
Epoch 10/10
188/188 [==============================] - 199s 1s/step - loss: 0.6932 - accuracy: 0.4995 - val_loss: 0.6931 - val_accuracy: 0.5040
```

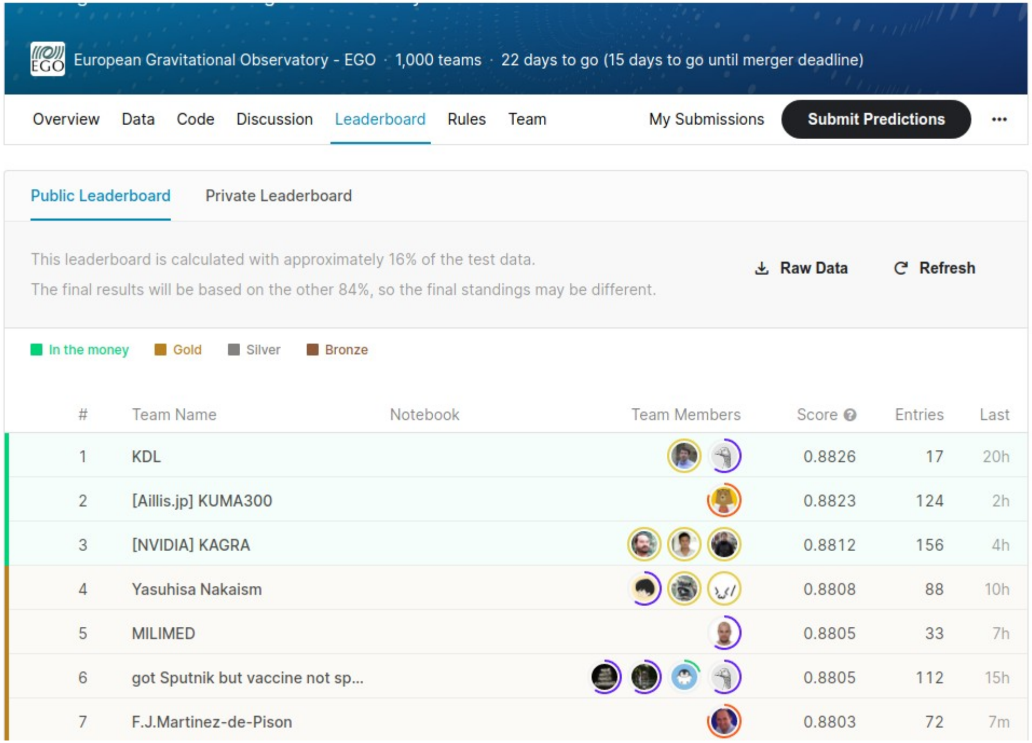
Backup

CWT/CNN Results

1000 examples, 80/20 train/test split

```
Model: "sequential_13"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_16 (Conv2D)           (None, 291, 88, 32)       3232
_________________________________________________________________
max_pooling2d_11 (MaxPooling (None, 48, 14, 32)        0
_________________________________________________________________
conv2d_17 (Conv2D)           (None, 46, 12, 64)        18496
_________________________________________________________________
max_pooling2d_12 (MaxPooling (None, 23, 6, 64)         0
_________________________________________________________________
conv2d_18 (Conv2D)           (None, 21, 4, 64)         36928
_________________________________________________________________
flatten_8 (Flatten)          (None, 5376)              0
_________________________________________________________________
dense_16 (Dense)             (None, 64)                344128
_________________________________________________________________
dense_17 (Dense)             (None, 2)                 130
=================================================================
Total params: 402,914
Trainable params: 402,914
Non-trainable params: 0
_________________________________________________________________
Epoch 1/10
3/3 [==============================] - 1s 107ms/step - loss: 0.9183 - accuracy: 0.3750
Epoch 2/10
3/3 [==============================] - 0s 23ms/step - loss: 0.7564 - accuracy: 0.5500
Epoch 3/10
3/3 [==============================] - 0s 25ms/step - loss: 0.6912 - accuracy: 0.5875
Epoch 4/10
3/3 [==============================] - 0s 24ms/step - loss: 0.6665 - accuracy: 0.6250
Epoch 5/10
3/3 [==============================] - 0s 24ms/step - loss: 0.6587 - accuracy: 0.6375
Epoch 6/10
3/3 [==============================] - 0s 28ms/step - loss: 0.6100 - accuracy: 0.6625
Epoch 7/10
3/3 [==============================] - 0s 24ms/step - loss: 0.5969 - accuracy: 0.6875
Epoch 8/10
3/3 [==============================] - 0s 23ms/step - loss: 0.5981 - accuracy: 0.6875
Epoch 9/10
3/3 [==============================] - 0s 25ms/step - loss: 0.5399 - accuracy: 0.7125
Epoch 10/10
3/3 [==============================] - 0s 24ms/step - loss: 0.5107 - accuracy: 0.6875
```

Leaderboard [as of 2021-09-08]



European Gravitational Observatory - EGO · 1,000 teams · 22 days to go (15 days to go until merger deadline)

This leaderboard is calculated with approximately 16% of the test data.

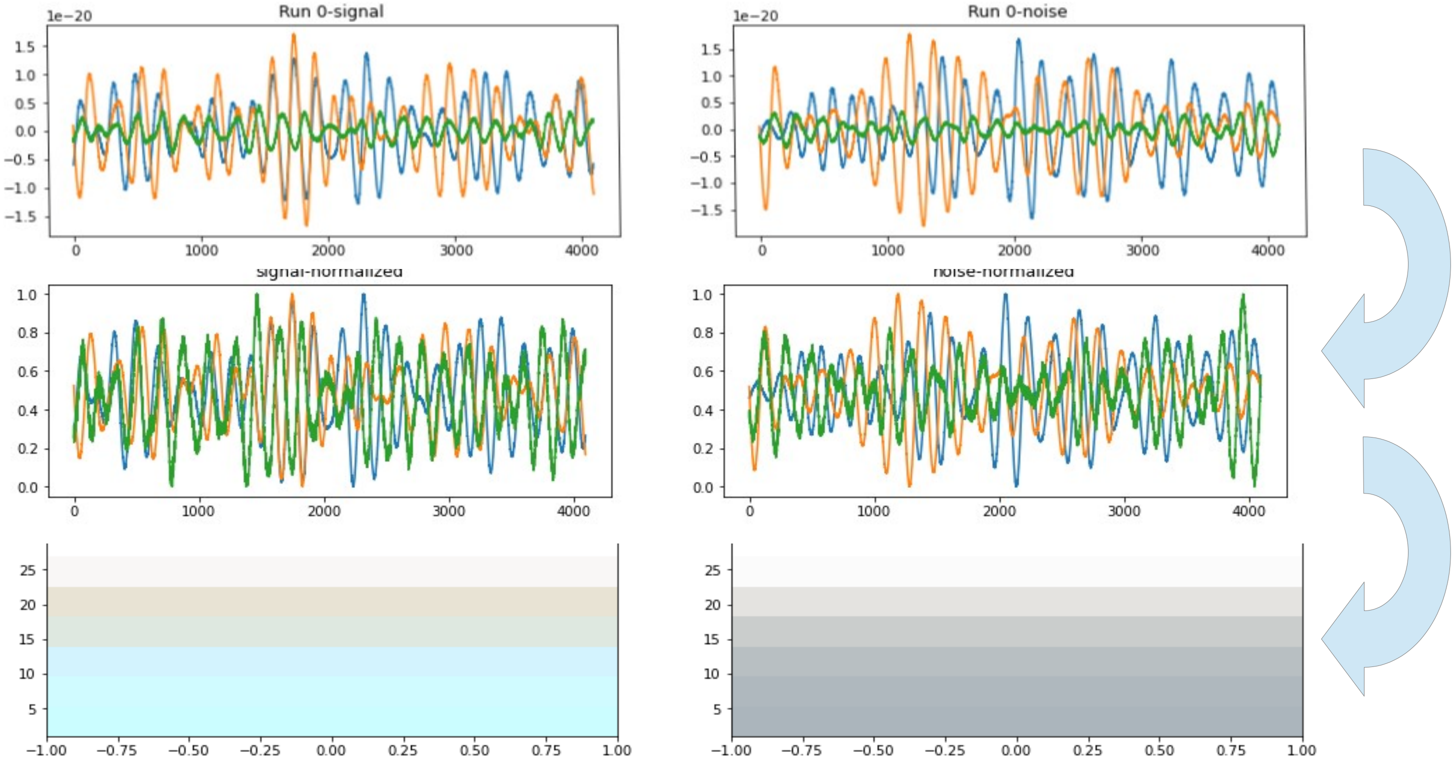
The final results will be based on the other 84%, so the final standings may be different.

■ In the money ■ Gold ■ Silver ■ Bronze

#	Team Name	Notebook	Team Members	Score	Entries	Last
1	KDL			0.8826	17	20h
2	[Aillis.jp] KUMA300			0.8823	124	2h
3	[NVIDIA] KAGRA			0.8812	156	4h
4	Yasuhisa Nakaism			0.8808	88	10h
5	MILIMED			0.8805	33	7h
6	got Sputnik but vaccine not sp...			0.8805	112	15h
7	F.J.Martinez-de-Pison			0.8803	72	7m

Continuous Wavelet Transform

- Transform detector time series into spectrograms (images)
- Train CNN with spectrogram training examples

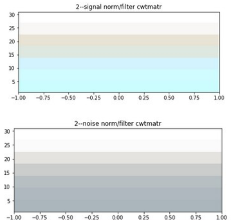


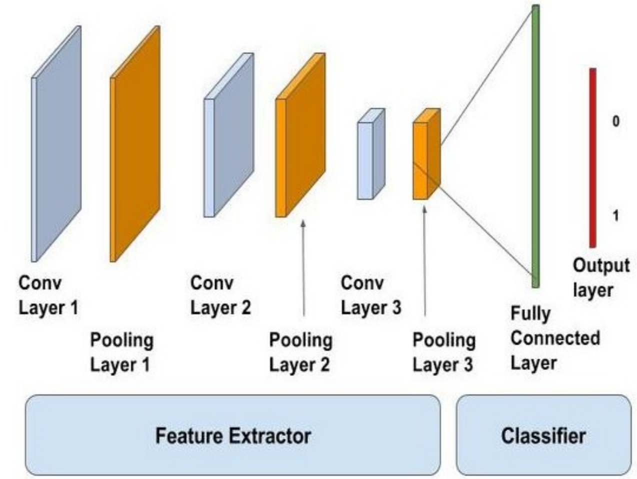
Normalize signals
(z transform)

Perform CWT transform
Apply bandpass filter

Convolutional Neural Net

Labeled
Spectrograms





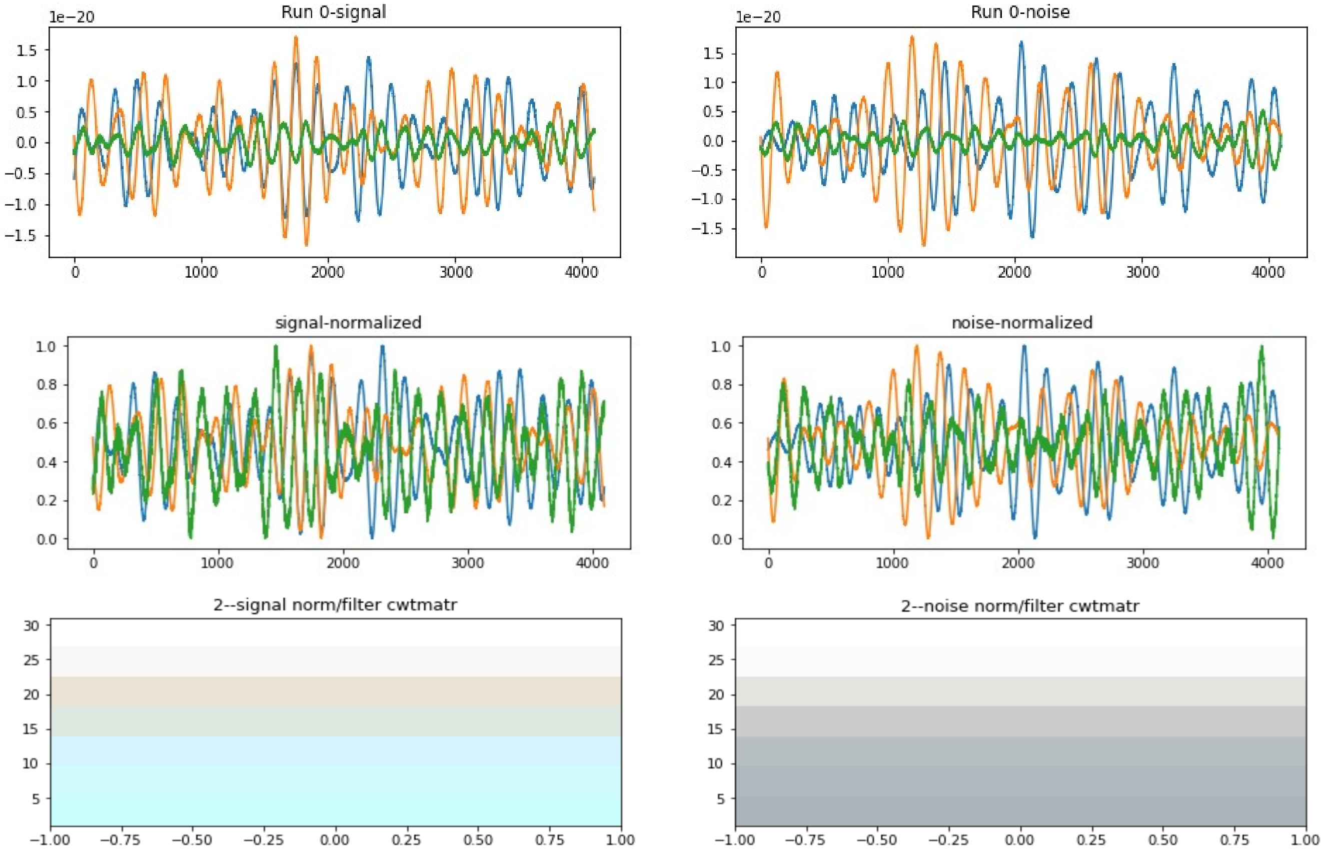
```
model=Sequential()
model.add(Conv2D(32,(3,3),activation='relu',input_shape=(21,4,1)))
model.add(MaxPool2D(2,2))
model.add(Flatten())
model.add(Dense(100,activation='relu'))
model.add(Dense(2,activation='softmax'))
model.compile(loss='sparse_categorical_crossentropy',optimizer='adam',metrics=['accuracy'])
```

https://www.analyticsvidhya.com/blog/2021/08/beginners-guide-to-convolutional-neural-network-with-implementation-in-python/s

CWT/CNN Results

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 19, 2, 32)         320
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 9, 1, 32)          0
_________________________________________________________________
flatten (Flatten)            (None, 288)               0
_________________________________________________________________
dense (Dense)                (None, 100)               28900
_________________________________________________________________
dense_1 (Dense)              (None, 2)                 202
=================================================================
Total params: 29,422
Trainable params: 29,422
Non-trainable params: 0
_________________________________________________________________
Epoch 1/10
250/250 [==============================] - 2s 3ms/step - loss: 0.6943 - accuracy: 0.5056
Epoch 2/10
250/250 [==============================] - 1s 3ms/step - loss: 0.6937 - accuracy: 0.5006
Epoch 3/10
250/250 [==============================] - 1s 3ms/step - loss: 0.6933 - accuracy: 0.5046
Epoch 4/10
250/250 [==============================] - 1s 3ms/step - loss: 0.6933 - accuracy: 0.5048
Epoch 5/10
250/250 [==============================] - 1s 3ms/step - loss: 0.6931 - accuracy: 0.5001
Epoch 6/10
250/250 [==============================] - 1s 3ms/step - loss: 0.6931 - accuracy: 0.4969
Epoch 7/10
250/250 [==============================] - 1s 3ms/step - loss: 0.6931 - accuracy: 0.4969
Epoch 8/10
250/250 [==============================] - 1s 3ms/step - loss: 0.6932 - accuracy: 0.5027
Epoch 9/10
250/250 [==============================] - 1s 3ms/step - loss: 0.6929 - accuracy: 0.4967
Epoch 10/10
250/250 [==============================] - 1s 3ms/step - loss: 0.6929 - accuracy: 0.5041
```

Creating Spectrogram


Run 0-signal
1e−20
Run 0-noise
1e−20
signal-normalized
noise-normalized
2--signal norm/filter cwtmatr
2--noise norm/filter cwtmatr