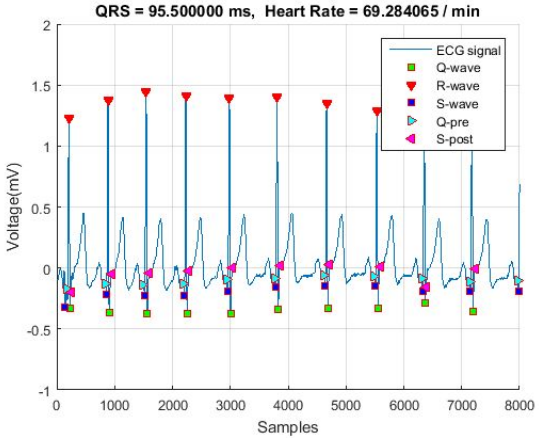
The Application of Deep Convolutional Networks for the Classification of ECG Signal



Edward Mina
Professor Byron Wallace Ph.D.
Northeastern University College of Engineering
Boston MA

ECG Heartbeat Classification: A Deep Transferable Representation

Mohammad Kachuee, Shayan Fazeli, Majid Sarrafzadeh
University of California, Los Angeles (UCLA)
Los Angeles, USA

Work	Approach	Average Accuracy (%)
This Paper	**Deep residual CNN**	**93.4**
Acharya *et al.* [23]	Augmentation + CNN	93.5
Martis *et al.* [24]	DWT + SVM	93.8
Li *et al.* [25]	DWT + random forest	94.6

< 4% of data

IV. Results

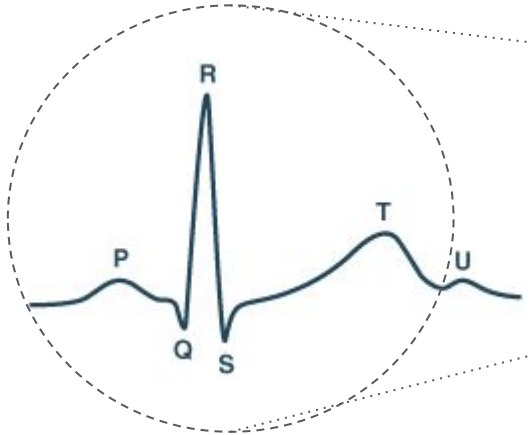
A. *Arrhythmia Classification and learning the representation*

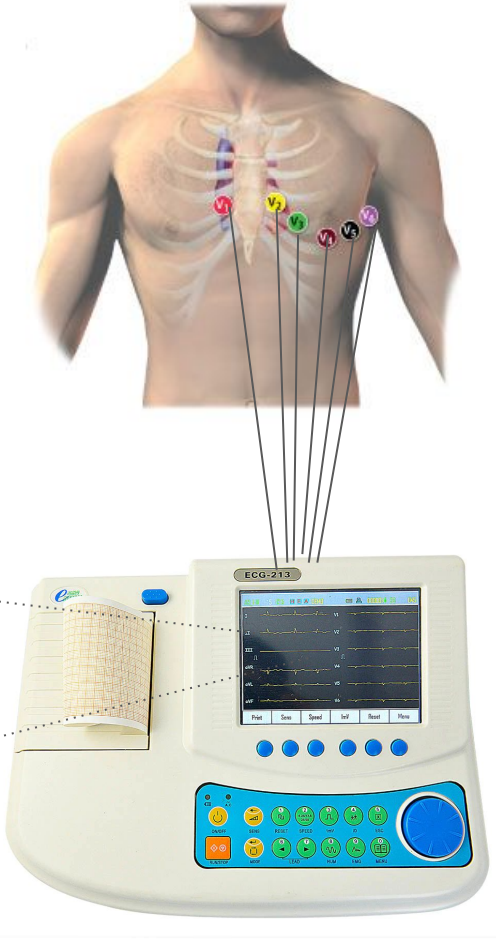
We evaluated the arrhythmia classifier of Section III-B on 4079 heartbeats (about 819 from each class) that are not used

- Performs Very Poorly with a more appropriately split dataset!
- 80/20 or 75/25

What is a heart beat?

- Measured Electrocardiogram (ECG)
- P wave = depolarization of the atria
- QRS = depolarization of ventricles
- T = T wave, which indicates Ventricular repolarization





Data

- ECG recordings of 48 patients at Beth-Israel Hospital.
- 30 minutes at 360 sample/sec
    - 650k pts per patient

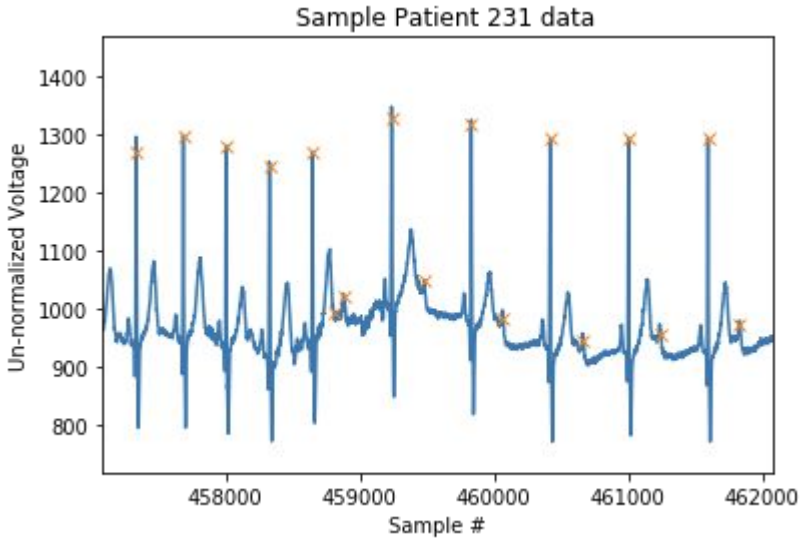


Figure 1: Raw ECG data for patient 231 with labelled peaks with respect to sample number and unnormalized voltage recording.

Labeled Heart Beats | Signal Data

	sample_num	type	aux
0	18	+	(N
1	111	L	NaN
2	343	L	NaN
3	571	L	NaN
4	807	L	NaN

	signal
0	955
1	955
2	955
3	955
4	955

MIT-BIH Arrhythmia Database

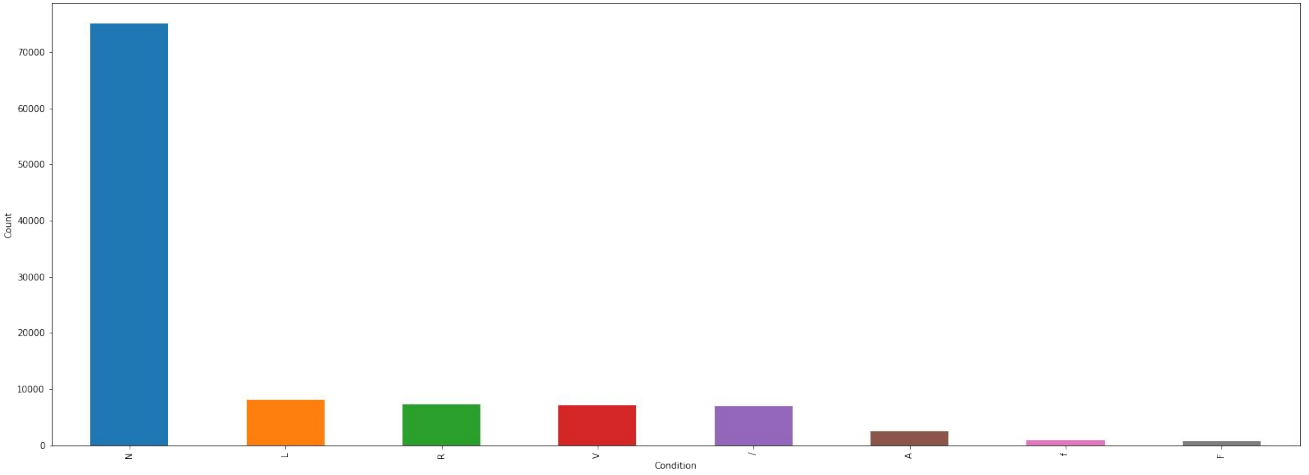
This database is described in

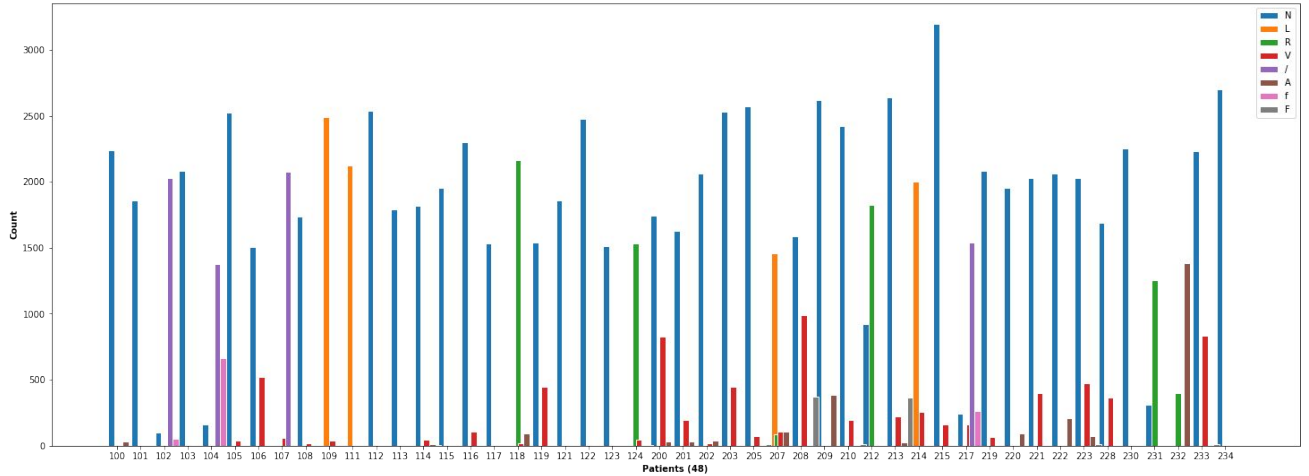
Moody GB, Mark RG. The impact of the MIT-BIH Arrhythmia Database. *IEEE Eng in Med and Biol* 20(3):45-50 (May-June 2001). (PMID: 11446209)

Please cite this publication when referencing this material, and also include the standard citation for PhysioNet:

Goldberger AL, Amaral LAN, Glass L, Hausdorff JM, Ivanov PCh, Mark RG, Mietus JE, Moody GB, Peng C-K, Stanley HE. PhysioBank, PhysioToolkit, and PhysioNet: Components of a New Research Resource for Complex Physiologic Signals. *Circulation* **101**(23):e215-e220 [Circulation Electronic Pages; http://circ.ahajournals.org/content/101/23/e215.full]; 2000 (June 13).

Class Imbalance





Class Simplification and Reduction

```
{'N': 'Normal beat',
 'L': 'Left bundle branch block beat',
 'R': 'Right bundle branch block beat',
 'A': 'Atrial premature beat',
 'a': 'Aberrated atrial premature beat',
 'J': 'Nodal (junctional) premature beat',
 'S': 'Supraventricular premature beat',
 'V': 'Premature ventricular contraction',
 'F': 'Fusion of ventricular and normal beat',
 '[': 'Start of ventricular flutter/fibrillation',
 '!': 'Ventricular flutter wave',
 ']': 'End of ventricular flutter/fibrillation',
 'e': 'Atrial escape beat',
 'j': 'Nodal (junctional) escape beat',
 'E': 'Ventricular escape beat',
 '/': 'Paced beat',
 'f': 'Fusion of paced and normal beat',
 'x': 'Non-conducted P-wave (blocked APB)',
 'Q': 'Unclassifiable beat',
 '|': 'Isolated QRS-like artifact'}
```

```
For class: N (Normal beat)
  (N) Normal beat
  (L) Left bundle branch block beat
  (R) Right bundle branch block beat
  (e) Atrial escape beat
  (j) Nodal (junctional) escape beat
For class: S (Supraventricular premature beat)
  (S) Supraventricular premature beat
  (A) Atrial premature beat
  (a) Aberrated atrial premature beat
  (J) Nodal (junctional) premature beat
For class: V (Premature ventricular contraction)
  (V) Premature ventricular contraction
  (E) Ventricular escape beat
For class: F (Fusion of ventricular and normal bea
  (F) Fusion of ventricular and normal beat
For class: Q (Unclassifiable beat)
  (/) Paced beat
  (Q) Unclassifiable beat
  (f) Fusion of paced and normal beat
```

Normalization

PATIENT: 207

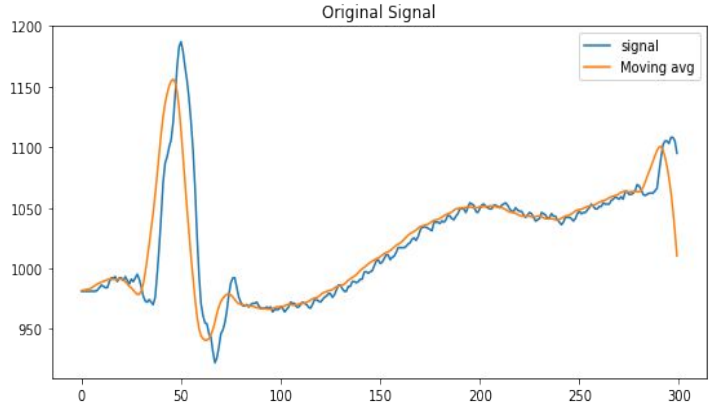
Norm 1: Creates a mean of 0, and a std of 1

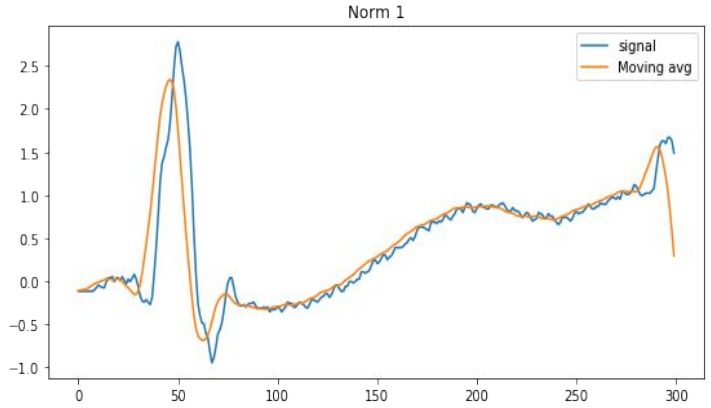
Norm 2: Creates a mean of 0, and y range from [-1 1]

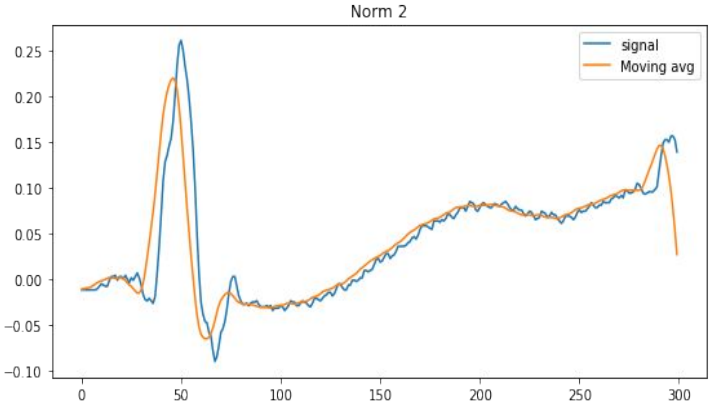
Norm 3: Creates a y range from [0 1]

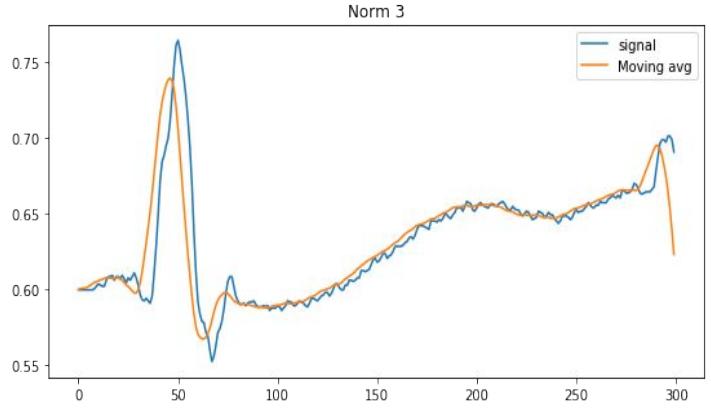
Selected Norm = 3

$$z_i = \frac{x_i - min(x)}{max(x) - min(x)}$$



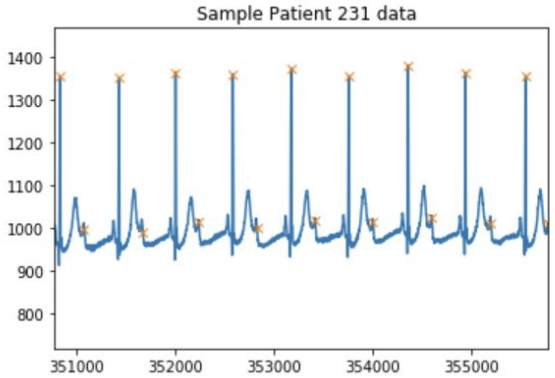




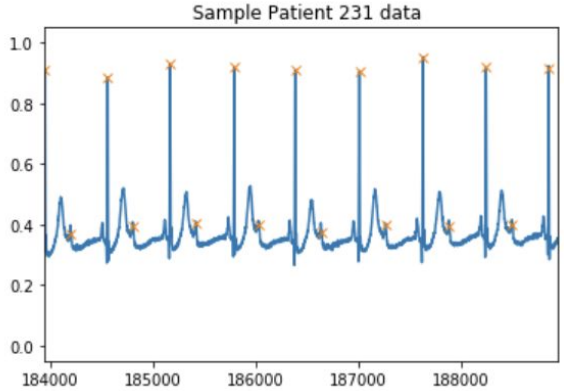


Effect of Normalization

```
1 hb.get_patient_data(patient=231,norm=False,sample_plot=True)
```

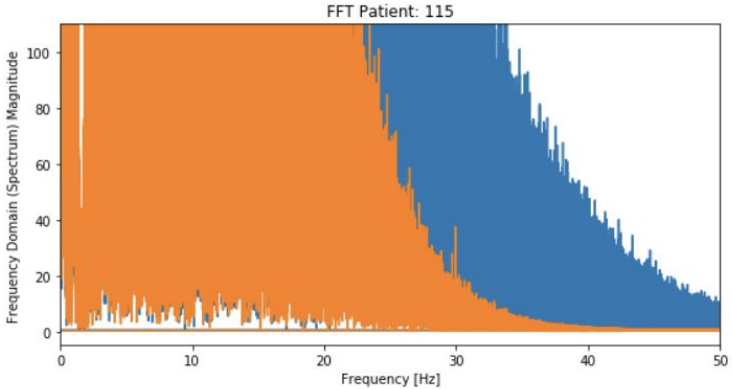


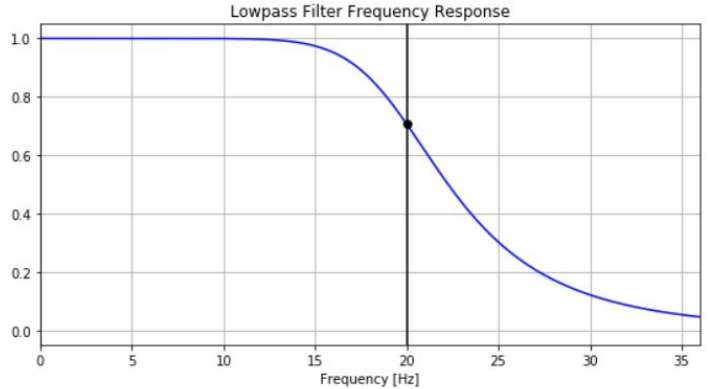

```
1 hb.get_patient_data(patient=231,norm=True,sample_plot=True)
```

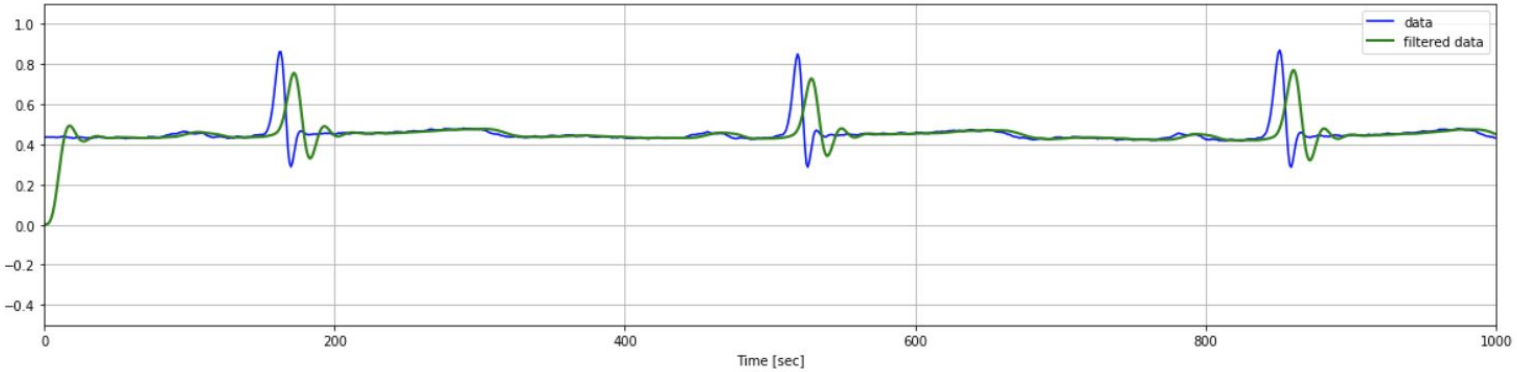


Further Signal Processing...

```
1 from normalizer import filt  #class of various filteration functions
2 patient= random.choice(hb.all_patients())
3 filt().low_pass_filter_plot(patient=patient,cutoff=20,fs=360)
```







Heart Beat Isolation Algo

Given Ecg Signal:

Find peaks w/r sample # (Christov)

$$Beat_i = \text{signal}\left[p_i - \frac{p_i - p_{i-1}}{2} : p_i + \frac{p_{i+1} - p_i}{2}\right], where\ p = peak$$

Zero_pad w/r max len

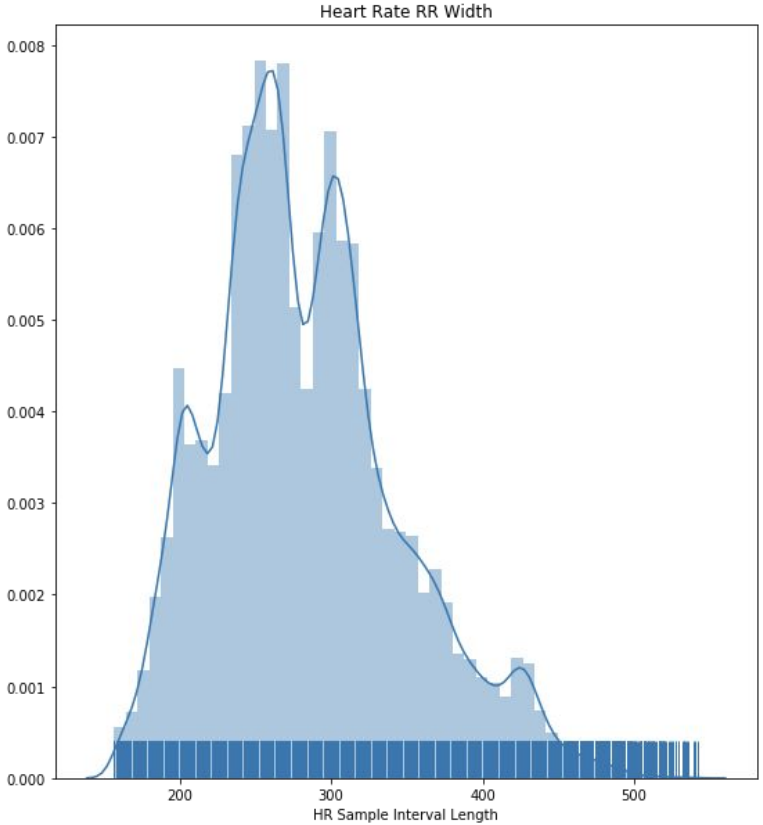
```
#isolation algorithim heart beat data
X,y,isolated_beat= hb.isolate_patient_data(patients=hb.all_patients(),classes=classes,
                    classes_further=hb.classes_further, classes_reducer=classes_reducer,
                     min_HR= 40,max_HR= 140,fs=360,verbose=False,plot_figs=True)
```

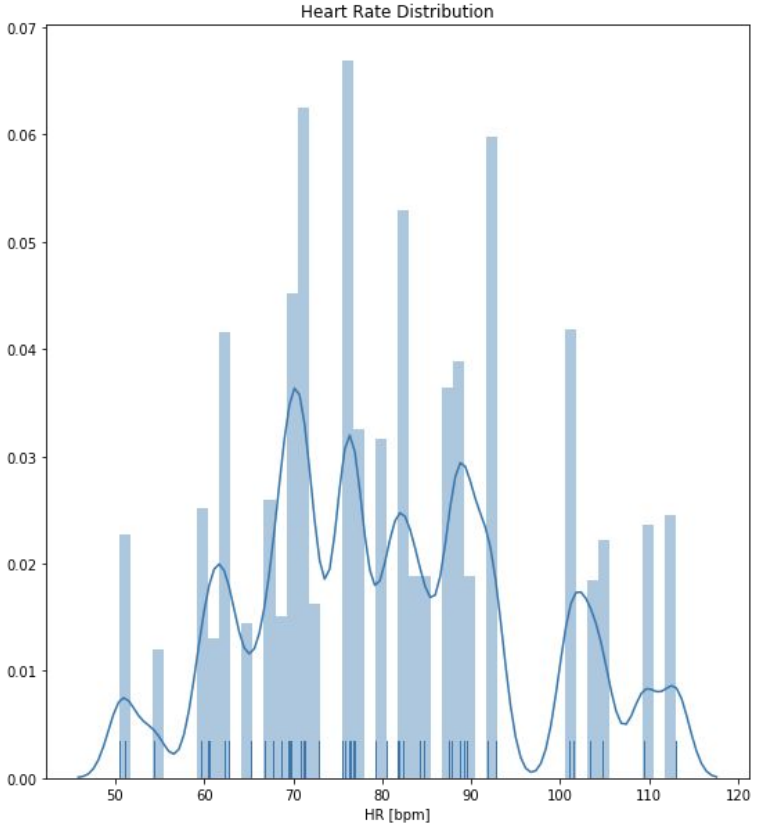
```
Examining 48 patients...
Percent: [>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>] 100.00% Done...

Padding...


Average HR Sample Len: 284.39 samples (0.79s per beat)
Average HR: 80.61 bpm
```

```
print("X shape:", X.shape)
print("y shape:", y.shape)
print("sample y vals:: [patient#, HR, Condition Class]:",y[0])
```

```
X shape: (107726, 539)
y shape: (107726, 3)
sample y vals:: [patient#, HR, Condition Class]: ['100' '75.536676138855' 'N']
```

Isolated HB Distribution

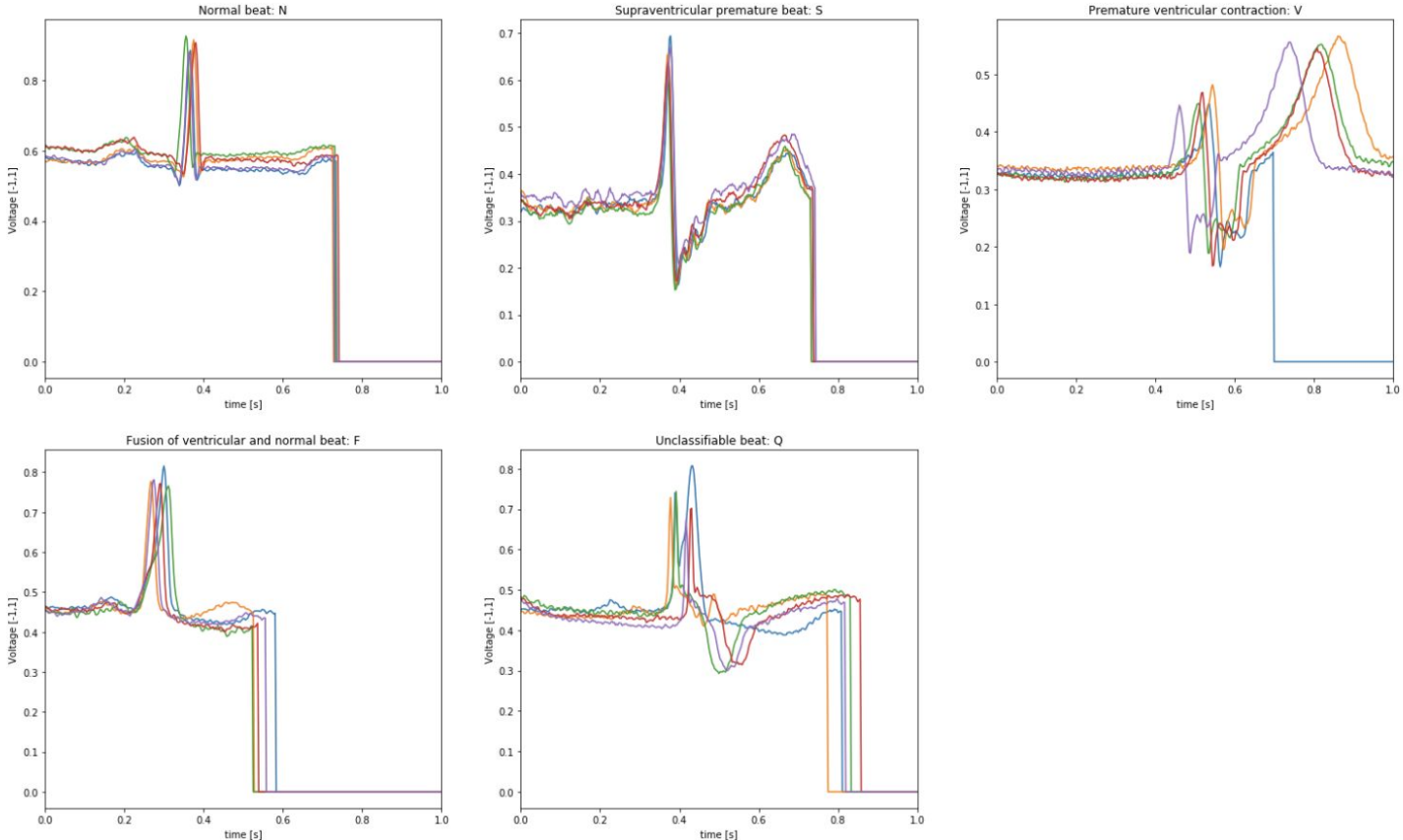




Heat Beat Classes

```
1 hb.show_sample_plots(X=X,y=y,classes=classes,classes_further=hb.classes_further,plot_xlim=1,dims=[2,3])
```

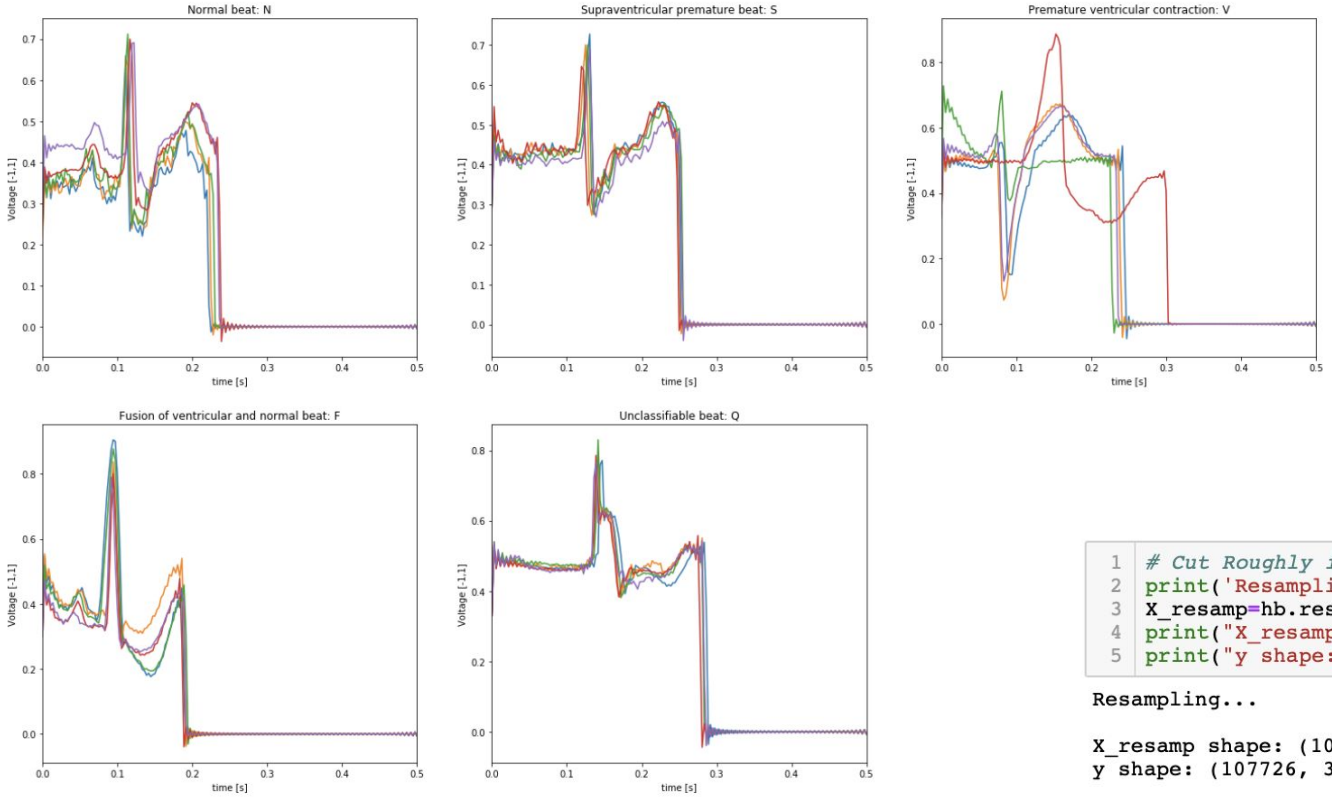
```
MAX HB TIME: 1.4972222222222222
```



Resampling 360Hz → 125Hz ~65% Reduction

```
hb.show_sample_plots(X=X_resamp,y=y,classes=classes,classes_further=hb.classes_further,
                     plot_xlim=.5,dims=[2,3])
```

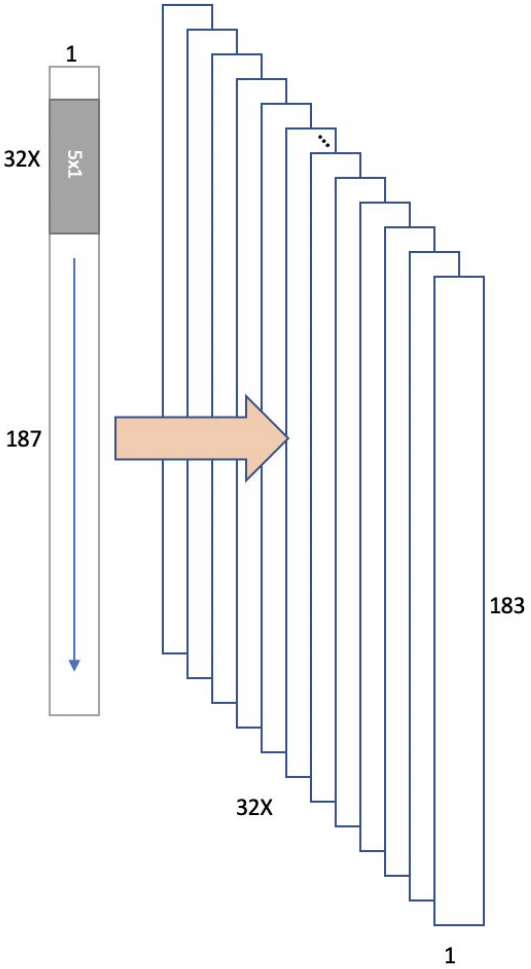
```
MAX HB TIME: 0.5194444444444445
```




```
# Cut Roughly in Half
print('Resampling...\n')
X_resamp=hb.resample_vals(X,samp_len=187)
print("X_resamp shape:", X_resamp.shape)
print("y shape:", y.shape)
```

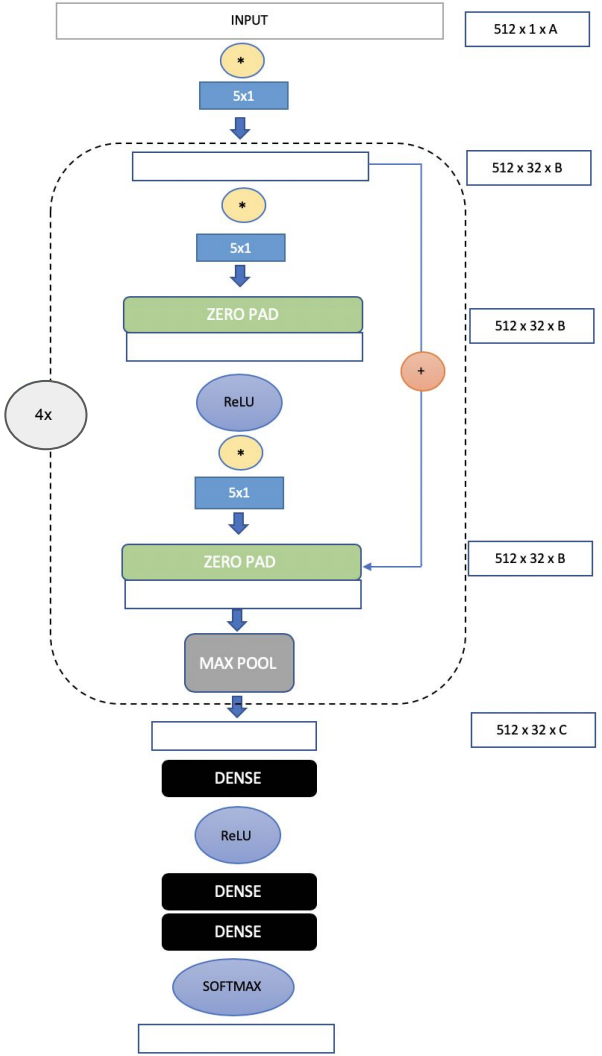
```
Resampling...

X_resamp shape: (107726, 187)
y shape: (107726, 3)
```

1D CNN



- Effective when examining features from shorter (fixed-length) segments of the data

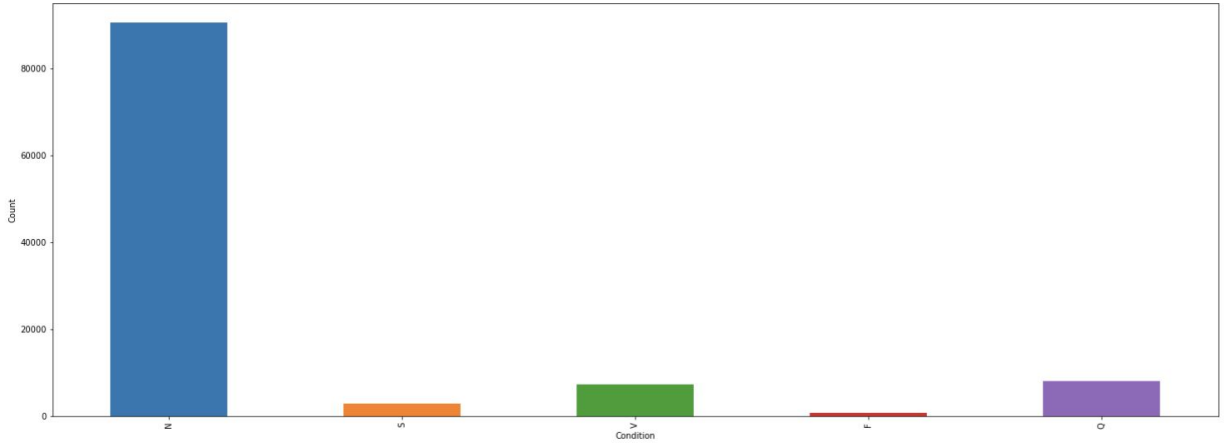
- 75% Training + 5% Validation
- 20% Test
  - Forced to have all classes present

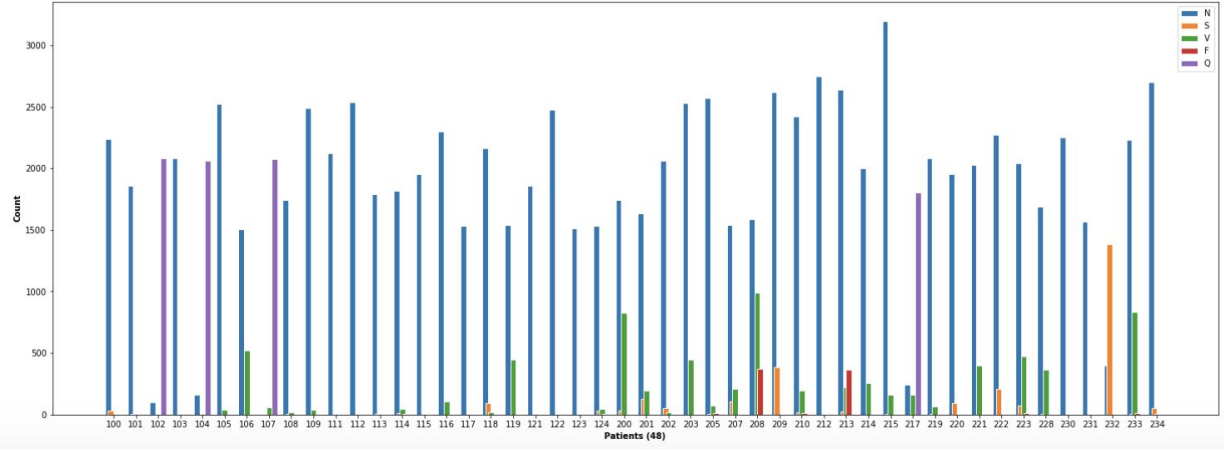


Classes Still Unbalanced…

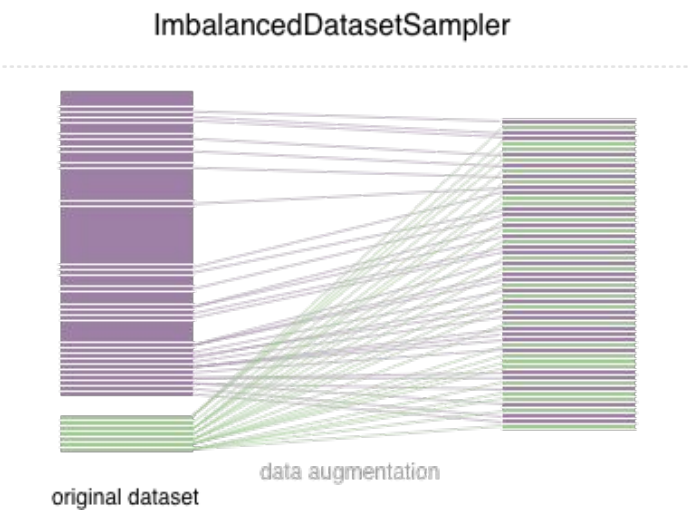
```
patient_dic=hb.distribution_bar(patients=hb.all_patients(),classes=classes,
                                classes_reducer=classes_reducer)
```

```
Generating_plot(s)...
```





80% Improvement With Unbalanced Class Methods




```
[135]  1 for hb,labels in dataloader['train']:
       2   for hb_index,label in enumerate(labels):
       3     print(hb_index,hb[hb_index].size(),label.cpu().numpy().shape,Counter(label.cpu()
```

```
 0 torch.Size([512, 187]) (512,) Counter({0: 106, 4: 105, 2: 104, 1: 102, 3: 95})
 1 torch.Size([512, 187]) (512,) Counter({3: 110, 2: 106, 4: 104, 1: 97, 0: 95})
 2 torch.Size([512, 187]) (512,) Counter({4: 114, 3: 108, 2: 106, 1: 93, 0: 91})
 3 torch.Size([512, 187]) (512,) Counter({4: 118, 2: 102, 3: 98, 1: 97, 0: 97})
 4 torch.Size([512, 187]) (512,) Counter({1: 110, 4: 102, 2: 102, 0: 101, 3: 97})
 5 torch.Size([512, 187]) (512,) Counter({0: 121, 2: 113, 1: 96, 3: 95, 4: 87})
 6 torch.Size([512, 187]) (512,) Counter({0: 123, 2: 114, 4: 95, 3: 93, 1: 87})
 7 torch.Size([512, 187]) (512,) Counter({4: 115, 3: 110, 0: 99, 2: 95, 1: 93})
 8 torch.Size([512, 187]) (512,) Counter({0: 118, 1: 101, 4: 100, 3: 97, 2: 96})
 9 torch.Size([512, 187]) (512,) Counter({4: 114, 1: 109, 3: 100, 2: 98, 0: 91})
10 torch.Size([512, 187]) (512,) Counter({0: 112, 2: 109, 4: 98, 3: 97, 1: 96})
11 torch.Size([512, 187]) (512,) Counter({1: 112, 0: 103, 3: 102, 4: 101, 2: 94})
```

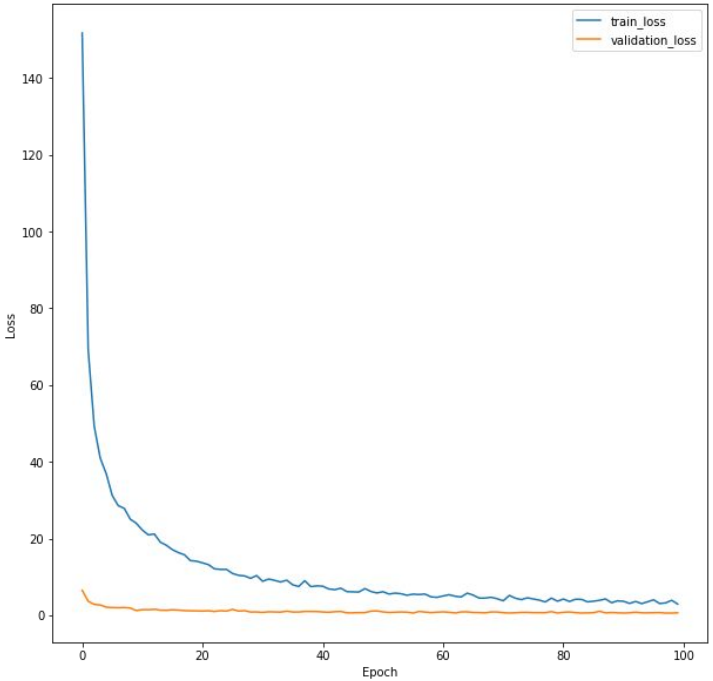
```
[160]  1 for hb,labels in dataloader['val']:
       2   for hb_index,label in enumerate(labels):
       3     print(hb_index,hb[hb_index].size(),label.cpu().numpy().shape,Counter(label.cpu()
```

```
0 torch.Size([512, 187]) (512,) Counter({0: 455, 4: 28, 2: 25, 1: 4})
1 torch.Size([512, 187]) (512,) Counter({0: 473, 4: 18, 2: 18, 1: 3})
2 torch.Size([512, 187]) (512,) Counter({0: 458, 2: 28, 4: 20, 1: 6})
3 torch.Size([512, 187]) (512,) Counter({0: 459, 4: 24, 2: 24, 1: 5})
4 torch.Size([512, 187]) (512,) Counter({0: 472, 2: 21, 4: 17, 1: 2})
5 torch.Size([512, 187]) (512,) Counter({0: 466, 4: 26, 2: 19, 1: 1})
```

General Observations

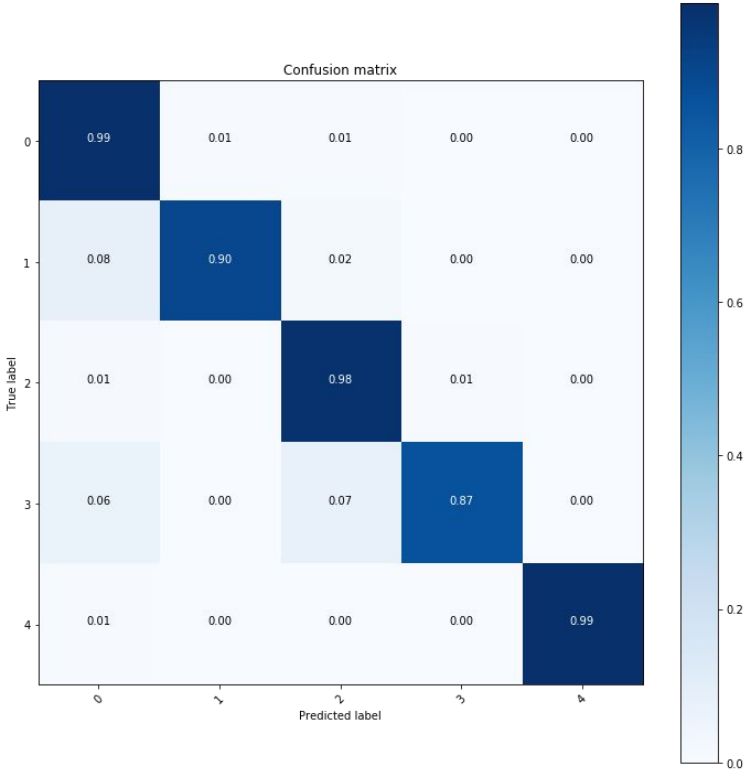
- Best filter size 32 x 5 x 1
  - Larger Values let to validation overfitting
- Best number of “blocks” 4 ~ 5
  - Relatively ineffective after 4
  - If <, lower performance
- Number of Epochs ~ 100
- Batch 512 (as per specified in the paper)
  - Poor loss behavior 128 < & < 1024
- Lesser Performance with Drop Out

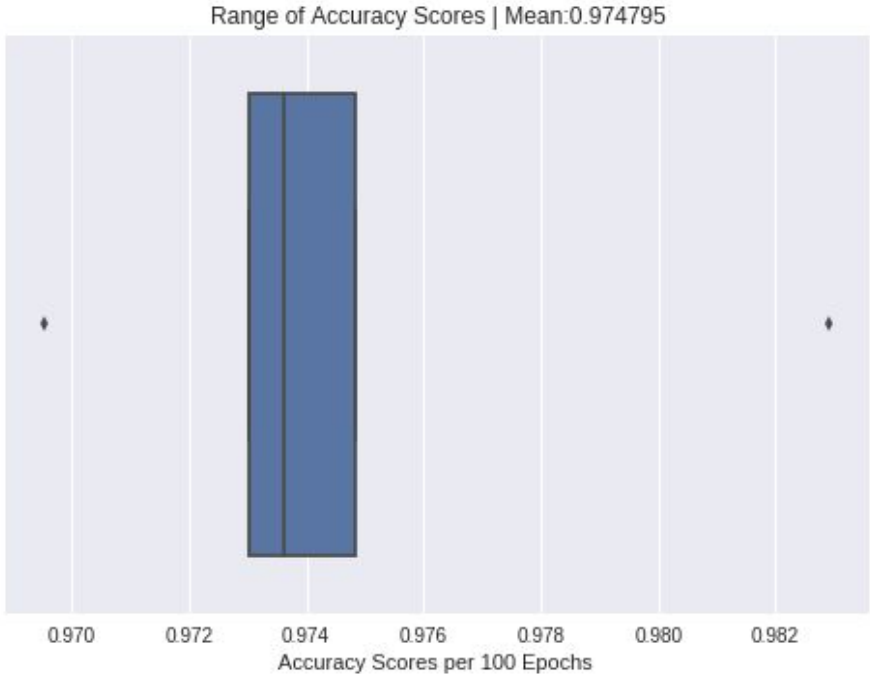
Results




```
Evaluating....
TEST ACC: 0.9831499791115443
              precision    recall  f1-score   support

           0       1.00      0.99      0.99     17939
           1       0.82      0.90      0.85       477
           2       0.90      0.98      0.94      1365
           3       0.76      0.87      0.81       157
           4       0.99      0.99      0.99      1605

   micro avg       0.98      0.98      0.98     21543
   macro avg       0.89      0.94      0.92     21543
weighted avg       0.98      0.98      0.98     21543
```

Results Continued...





futurework

- Different Possible Combinations of Classes
- Examination RNN’s
- Using bpm as a reference
- Isolate beats different or looking at Signal as whole
- Different Processing Methodology
  - Butterworth with different cutoffs
  - Signal smooth techniques
  - Normalizing methods

Questions?