Electrocardiogram Analysis for Heart Disease Anomaly Detections

Tiago Oliveira
@tiagoooliveira
~Solution Architect - Foxconn

Similar Solutions

- Academic researches {i.e. MIT/Physionet, Stanford/Andrew Ng}
- Prediction based on heart beat sound
- Prediction based on Wearables/Raw data
- Biggest challenge in healthcare-diagnosis: Accuracy, confidence - collecting signals and predicting

Dataset – Feature Engineering

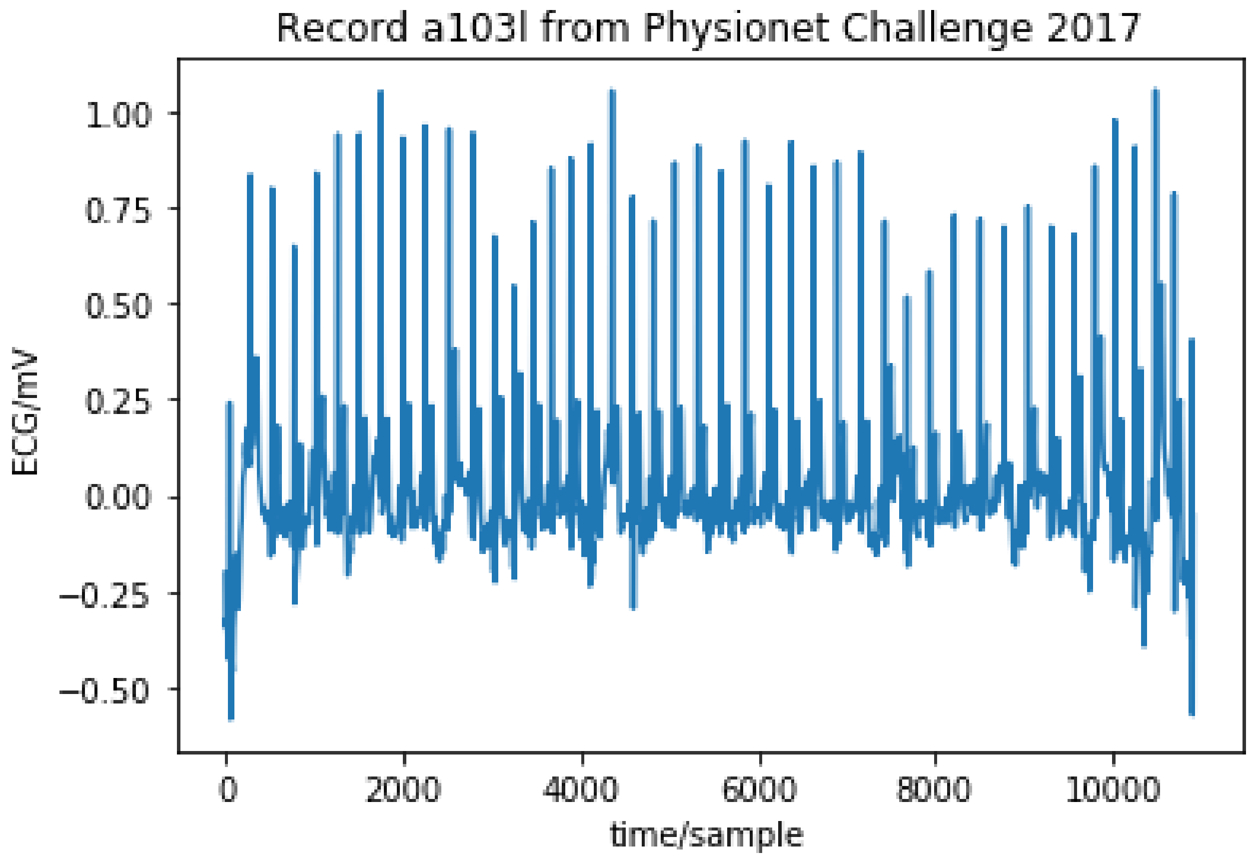
- Physionet Challenge: Consortium of universities and hospitals that provide real-world signals to be used as bases of training in challenges to improve diagnostic tools
- WFDB from PhysioBank: Wave Form Database - prevent reinventing the wheel
  - Easy way to interpret the wave form, without need to understand fourrier series, for example

An open access database for the evaluation of heart sound algorithms. (ncbi.nlm.nih.gov/pubmed)							
PhysioBank, PhysioToolkit, and PhysioNet | Components of a New Research Resource for Complex Physiologic Signals (circ.ahajournals.org/content/101/23/e215.full)							
	2010 Challenge	**2011 Challenge**	**2013 Challenge**	**2014 Challenge**	**2015 Challenge**	**2016 Challenge**	**2017 Challenge**
[fs] Frequency of the Record	125	500	1000	360	250	2000	300
[n_sig] Signals collected	6	12	4	6	4	1	1
[fmt] Bytes format	['16', '16', '16', '16', '16', '16']	['16','16','16','16','16','16','16','16','16','16','16','16']	['16','16','16','16']	['16', '16', '16', '16', '16', '16']	['16', '16', '16', '16']	['16']	['16']
[sig_len] Signal length – array	75000	5000	6000	206517	75000	243997	18260
[samps_per_frame] Samples of each channel present in each frame	[1, 1, 1, 1, 1, 1]	[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]	[1, 1, 1, 1]	[1, 1, 1, 1, 1, 1]	[1, 1, 1, 1]	[1]	[1]
[sig_name] Signal names. Used with sig_units to form y labels, if ylabel is not set	['RESP', 'ECG I', 'ECG V', 'ECG II', 'PLETH', 'ABP']	['I','II','III','aVR','aVF','aVL','V1','V2','V3','V4','V5', 'V6']	['AECG1', 'AECG2', 'AECG3', 'AECG4']	['ECG II','Pressure1','Pressure 2','Pressure','Pressure','No signal']	['II', 'V', 'PLETH', 'ABP']	['PCG']	['ECG']
[comments] Comments to be written to the header file	[]	['<age>: 0 <sex>: ?']	[]	[]	['Asystole', 'True alarm']	['Abnormal']	[]
[units] Units of each signal channel	['pm', 'mV', 'mV', 'mV', '?', 'mmHg']	['mV','mV','mV','mV','mV','mV','mV','mV','mV','mV','mV','mV']	['uV', 'uV', 'uV', 'uV']	['mV', 'mV', 'mV', 'mV', 'mV', 'mV']	['mV', 'mV', 'NU', 'mmHg']	['mV']	['mV']
Training Samples	100	1500	75	100	750	2290	8044
Test Samples	200	39	100	0	0	0	0

Dataset – Feature Engineering

```
record = wfdb.rdrecord('../../Databases/physionet/training_2017/training/a103l', channels='all')
wfdb.plot_wfdb(record=record, title='Record a103l from Physionet Challenge 2017')
display(record.__dict__)
print(record.p_signal.shape)
for x in range(0, 30):
    print(record.p_signal[x])
```

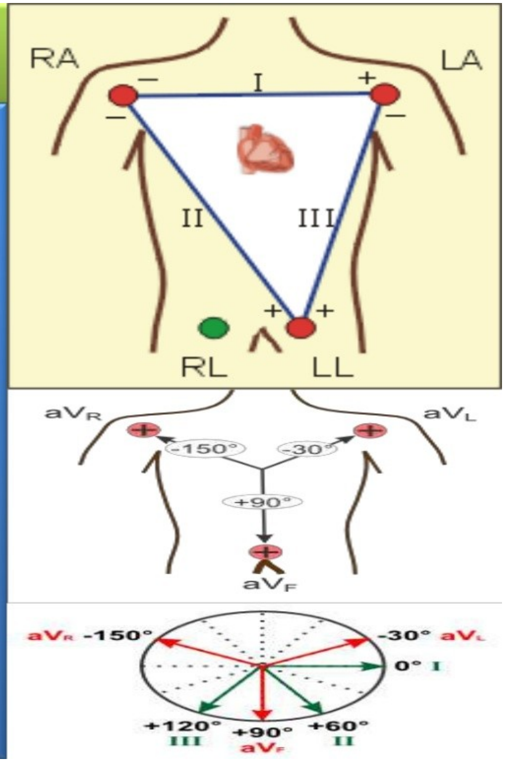
```
{'adc_gain': [1000.0],
 'adc_res': [16],
 'adc_zero': [0],
 'base_counter': None,
 'base_date': None,
 'base_time': '2013',
 'baseline': [0],
 'block_size': [0],
 'byte_offset': [24],
 'checksum': [0],
 'comments': [],
 'counter_freq': None,
 'd_signal': None,
 'e_d_signal': None,
 'e_p_signal': None,
 'file_name': ['a103l.mat'],
 'fmt': ['16'],
 'fs': 300,
 'init_value': [-202],
 'n_sig': 1,
 'p_signal': array([[-0.202],
        [-0.235],
        [-0.272],
        ...,
        [-0.313],
        [-0.173],
        [-0.052]]),
 'record_name': 'a103l',
 'samps_per_frame': [1],
 'sig_len': 10904,
 'sig_name': ['ECG'],
 'skew': [None],
 'units': ['mV']}
(10904, 1)
None
[-0.202]
[-0.235]
[-0.272]
[-0.305]
```



Dataset – Feature Engineering

ECG Leads

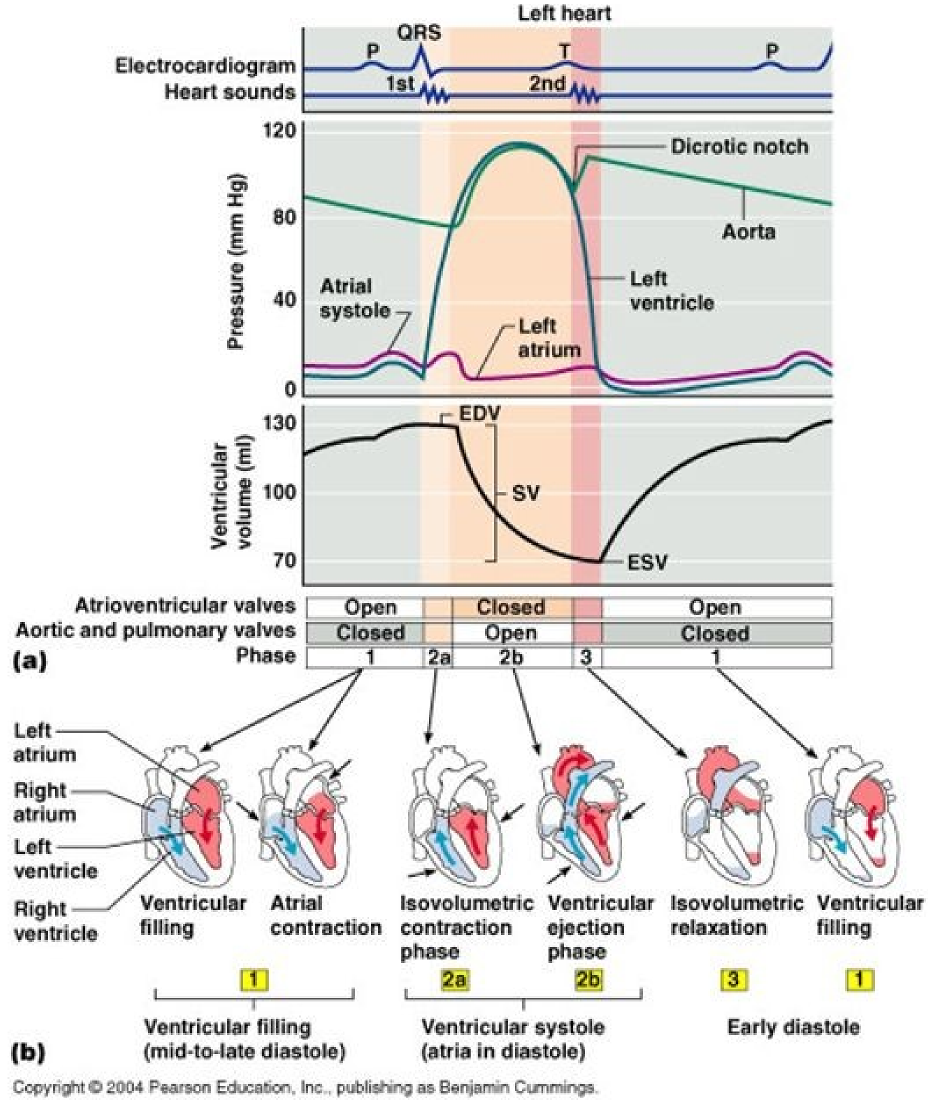
- The 12 lead consist of:
  - Three Standard limb or bipolar leads (I, II, III) utilize three electrodes; these leads form a triangle known as Einthoven's Triangle.
  - Three Augmented unipolar leads (aV_R, aV_L, aV_F).
  - Six Precordial unipolar leads (V_1, V_2, V_3, V_4, V_5, V_6).



Regional association with ECG

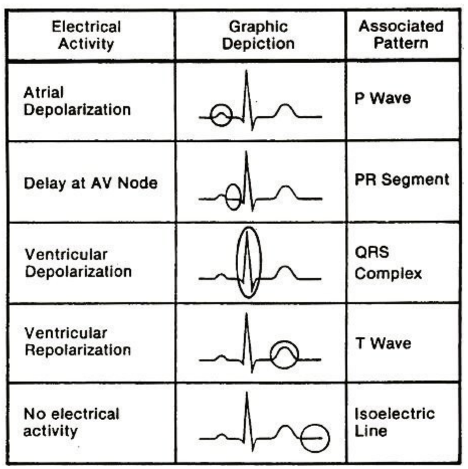
Area of infarction	Leads associated	Vessels involved
Inferior	Leads II, III, and aVF; ST elevations	Right coronary artery, left circumflex
Posterior	Leads V1, V2, V3 ST depression; large R wave	Proximal right coronary artery, left circumflex
Anterior	Leads V1, V2, V3, V4; ST elevation	Left anterior descending
Lateral	Leads V1, AVL, V5, V6; ST elevation	Left circumflex
Right ventricular	Elevations in leads II, III, aVF, and V1; elevation greater in III than II; large R wave V4	Proximal right coronary artery

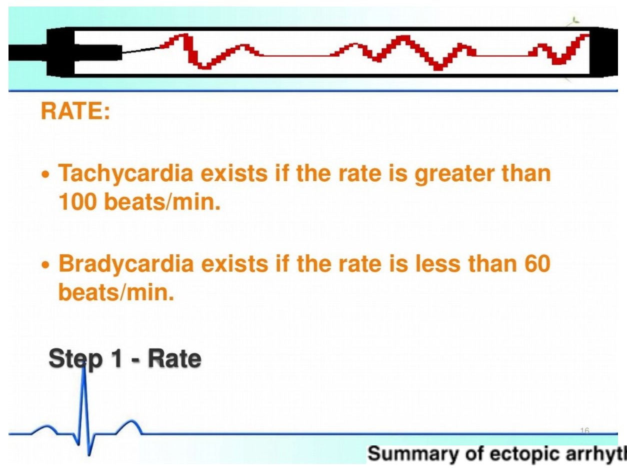
Cardiac Cycle

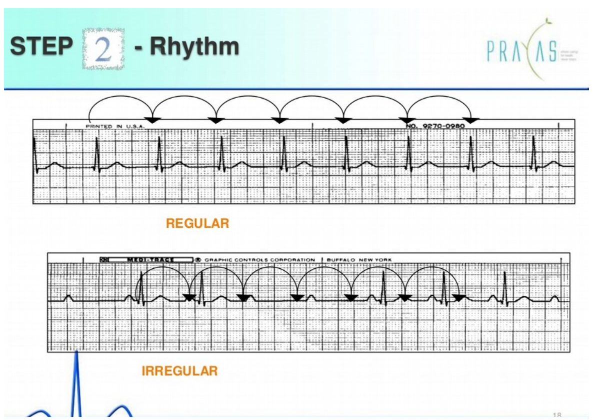


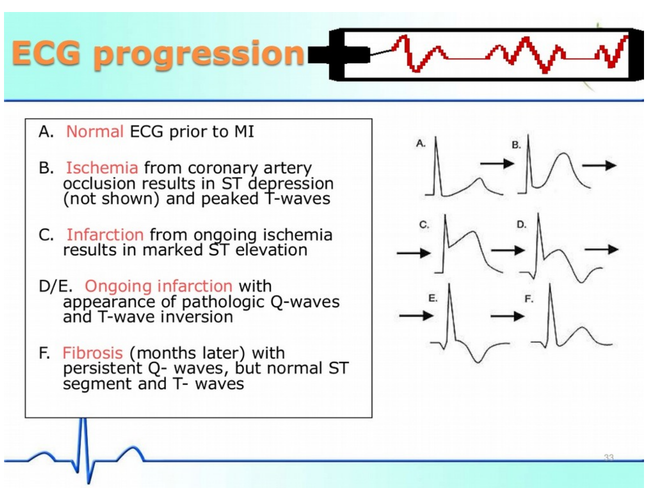
Dataset – Feature Engineering

- Common Patterns from ECG signal - Labeling
- Heart Electrical Events: Animation https://library.med.utah.edu/kw/pharm/hyper_heart1.html









Summary of ectopic arrhythmias

ARRHYTHMIA	ECTOPIC RATE	ECG DESCRIPTION
Wandering Atrial Pacemaker	60–100	Multiple P wave morphologies (usually 3 or more), variable rate
Ectopic Atrial Rhythm	40–250	Regular P waves with abnormal axis, PR interval > 120 msec, flat baseline between P waves, AV conduction may be 1:1 or variable
Mulitifocal Atrial Tachycardia	100–180	At least 3 P wave morphologies, varying PR intervals, rate > 100
Junctional Rhythms	40–120	Regular ventricular rhythm with P waves slightly before, hidden inside, or after QRS complex, PR interval < 120 msec
VT	120–250	Wide QRS tachycardia, regular ventricular rate

Model Architecture

- Unsupervised anomaly detection using Autoencoder
- Its goal is to induce a representation (encoding) for a set of data by learning an approximation of the identity function of this data
- Code snippet, Building Block Modeling

jupyter ECG - Autoencoders - Signal-Anomaly-Detection Last Checkpoint: a few seconds ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

In [1]:

```
import tensorflow as tf
import numpy as np
import matplotlib.pyplot as plt
import random

from keras.models import Sequential, Model
from keras.layers import LSTM, RepeatVector
```

```
/home/tiagoooliveira/anaconda3/lib/python3.6/site-packages/h5py/__init__.py:34: FutureWarning: Conversion of the sec
ond argument of issubdtype from `float` to `np.floating` is deprecated. In future, it will be treated as `np.float64
== np.dtype(float).type`.
  from ._conv import register_converters as _register_converters
Using TensorFlow backend.
```

In [2]:

```
#12-lead ECG recordings (leads I, II, II, aVR, aVL,aVF, V1, V2, V3, V4, V5, and V6)
#The leads are recorded simultaneously

#Simulating 12-leads signals
#normalized interval (-1) -> (1)
def ecg_gen(batch_size):
    seq_length = 10

    batch_x = []
    batch_y = []
    for _ in range(batch_size):
        rand = random.random() * 2 * np.pi

        lead_1 = np.sin(np.linspace(0.0 * np.pi + rand,
                                    3.0 * np.pi + rand, seq_length * 2))
        lead_2 = np.cos(np.linspace(0.0 * np.pi + rand,
                                    3.0 * np.pi + rand, seq_length * 2))

        lead_3 = np.sin(np.linspace(0.0 * np.pi + rand,
                                    3.0 * np.pi + rand, seq_length * 2))
        lead_aVR = np.cos(np.linspace(0.0 * np.pi + rand,
                                    3.0 * np.pi + rand, seq_length * 2))

        lead_aVL = np.sin(np.linspace(0.0 * np.pi + rand,
                                    3.0 * np.pi + rand, seq_length * 2))
        lead_aVF = np.cos(np.linspace(0.0 * np.pi + rand,
                                    3.0 * np.pi + rand, seq_length * 2))

        lead_V1 = np.sin(np.linspace(0.0 * np.pi + rand,
                                    3.0 * np.pi + rand, seq_length * 2))
        lead_V2 = np.cos(np.linspace(0.0 * np.pi + rand,
                                    3.0 * np.pi + rand, seq_length * 2))

        lead_V3 = np.sin(np.linspace(0.0 * np.pi + rand,
                                    3.0 * np.pi + rand, seq_length * 2))
        lead_V4 = np.cos(np.linspace(0.0 * np.pi + rand,
                                    3.0 * np.pi + rand, seq_length * 2))

        lead_V5 = np.sin(np.linspace(0.0 * np.pi + rand,
                                    3.0 * np.pi + rand, seq_length * 2))
```

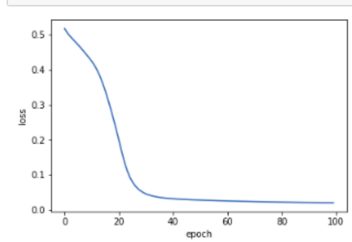
In [21]:

```
b_size = 5
n_epoch = 3
X_train, Y_train = ecg_gen(batch_size)

X_train

m = Sequential()

#Autoencoder
m.add(LSTM(12, input_shape=(10, 12)))
m.add(RepeatVector(10))
m.add(LSTM(12, return_sequences=True))

print(m.summary())

m.compile(loss='mse', optimizer='adam')

history = m.fit(X_train, Y_train, n_epoch, b_size)
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_15 (LSTM)               (None, 12)                1200
_________________________________________________________________
repeat_vector_8 (RepeatVecto (None, 10, 12)            0
_________________________________________________________________
lstm_16 (LSTM)               (None, 10, 12)            1200
=================================================================
Total params: 2,400
Trainable params: 2,400
Non-trainable params: 0
_________________________________________________________________
None
Epoch 1/5
10/10 [==============================] - 1s 97ms/step - loss: 0.4932
Epoch 2/5
10/10 [==============================] - 0s 4ms/step - loss: 0.4883
Epoch 3/5
10/10 [==============================] - 0s 4ms/step - loss: 0.4842
Epoch 4/5
10/10 [==============================] - 0s 4ms/step - loss: 0.4802
Epoch 5/5
10/10 [==============================] - 0s 4ms/step - loss: 0.4764
```

In [5]:

```
plt.plot(history.history['loss'])
plt.ylabel("loss")
plt.xlabel("epoch")
plt.show()
```



Event Driven Architecture

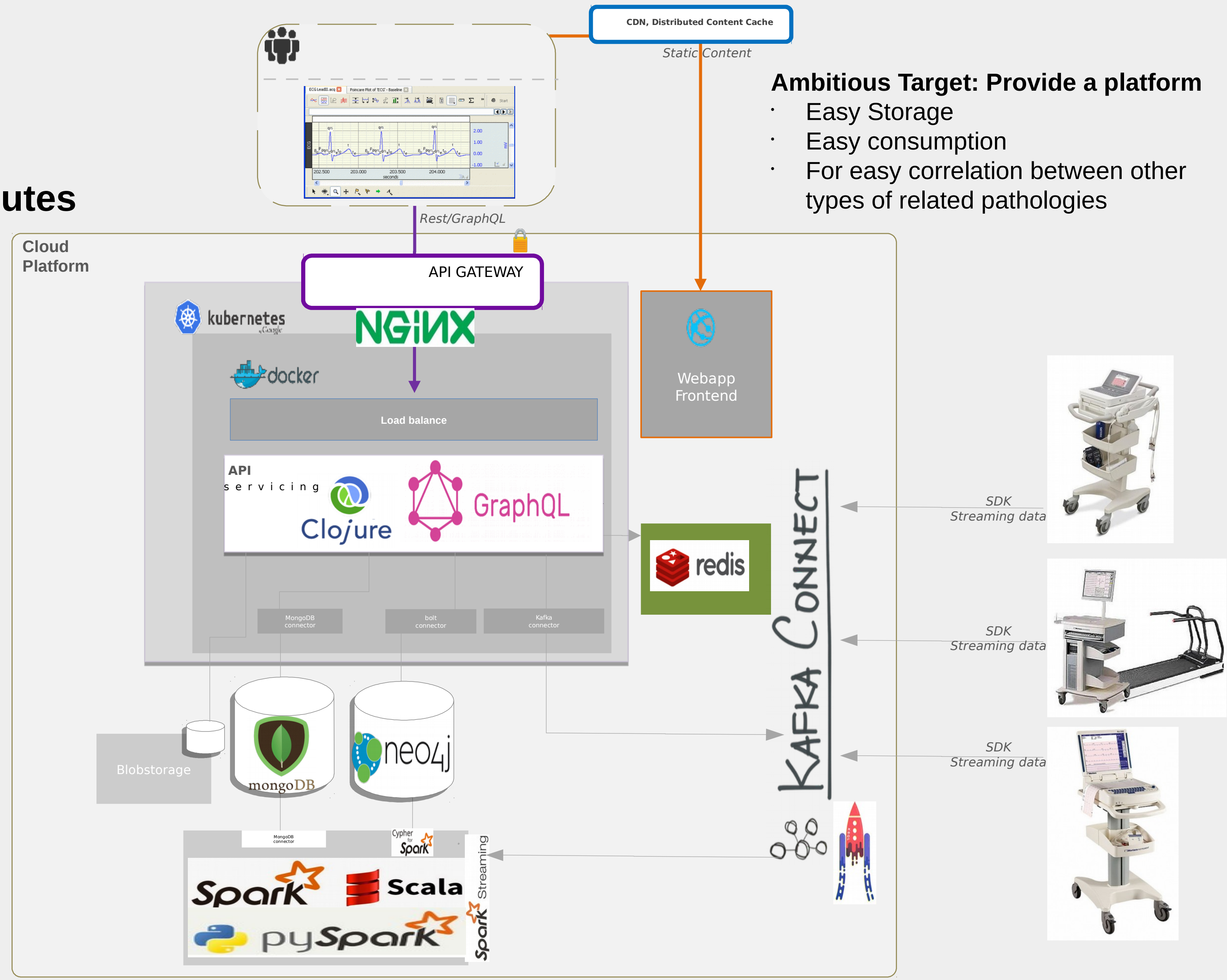
Application Architecture

Quality Attributes

- Extensibility
- Connectivity
- Performance
- Real-time AI/ML
- Real-time reports

Ambitious Target: Provide a platform

- Easy Storage
- Easy consumption
- For easy correlation between other types of related pathologies



Next Steps

- Hyperparameter Tunning - Autoencoders
- Structured Detection/Prediction, Grammar Graph (PyStruct, Cypher Pattern Matching)
- Benchmarking ML Algorithms Used by Similar Solutions/Researches: Hidden Markov Model, CNN, k-NN

Thank you

- https://github.com/TIAGOOOLIVEIRA/ECG-CVD-anomaly-detection
- https://www.linkedin.com/in/tiagoliveira/

- References

  - http://circ.ahajournals.org/content/101/23/e215.full Research Resource for Complex Physiologic Signals
  - http://wfdb.readthedocs.io/en/latest/index.html WFDB documentation
  - https://library.med.utah.edu/kw/pharm/hyper_heart1.html Heart electrical events - animation
  - http://ecg-interpretation.blogspot.de/2014/06/ecg-blog-92-basic-concepts-5-lvh.html
  - https://physionet.org/
  - https://www.slideshare.net/rohanchoudhari/ecg-23614046
  - https://arxiv.org/pdf/1707.01836.pdf Cardiologist-Level Arrhythmia Detection with Convolutional Neural Networks
  - http://www.internationaljournalofcardiology.com/article/S0167-5273(05)00404-3/abstract Common errors in computer electrocardiogram interpretation