intro-to-ecg

July 31, 2023

1 ECG Library

1.1 Suggested pipeline

1.1.1 Load ECG images

[4]:
```
import os
from PIL import Image
```

[5]:
```
dirname = 'examples/example_images/'
```

[6]:
```
images = os.listdir(dirname)
```

[7]:
```
images = sorted(images, key=lambda a: int(a.split('.')[0]))
```

[8]:
```
images
```

[8]:
```
['1.png',
 '2.png',
 '3.png',
 '4.png',
 '5.png',
 '6.png',
 '7.png',
 '8.png',
 '9.png',
 '10.png',
 '11.png',
 '12.png']
```

[9]:
```
images = [Image.open(dirname + filename) for filename in images]
```

[7]:
```
images[1]
```

[7]:

1.1.2 Convert images to signal

[2]:
```
import ECG.api as api
import numpy as np
```

[10]:
```
signal = [api.convert_image_to_signal(np.asarray(image)[:,:,:3]) for image in␣
↪images]
```

1.1.3 display explanations

[11]:
```
def display_text_explanation(explanation:api.TextExplanation):
    print('Text explanation:', explanation.content)

def display_text_and_image_explanation(explanation:api.TextAndImageExplanation):
    print('Text explanation:', explanation.text)
    print('GradCAM visualization:')
    return explanation.image
```

1.1.4 Get signal of all 12 ECG leads

Recommended sampling rate is 500

[12]:
```
sampling_rate = 500
```

[13]:
```
mm_per_mv = 10
```

[14]:
```
signal = np.asarray([i / mm_per_mv for i in signal])
```

[15]:
```
assert len(signal.shape) == 2
assert signal.shape[0] == 12
```

1.1.5 Check whether ST-elevation is present

[16]:
```
res = api.check_ST_elevation(signal, sampling_rate=sampling_rate)
print('Result:', res[0])
print('Explanation:')
display_text_explanation(res[1])
```

```
Result: ElevatedST.Abscent
Explanation:
Text explanation: ST elevation value in lead V3 (0.04126003416910934 mV) did not
exceed the threshold 0.2, therefore ST elevation was not detected.
```

[17]:
```
res = api.check_ST_elevation_with_NN(signal)
print('Result:', res[0])
print('Explanation:')
display_text_and_image_explanation(res[1])
```

```
Load model at ./ECG/NN_based_approach/Models/Conv1_ste_model.pt
Result: ElevatedST.Abscent
Explanation:
Text explanation: Significant ST elevation probability is 0.3325
GradCAM visualization:
```

[17]:

1.1.6 Evaluate risk markers

[18]:
```
api.evaluate_risk_markers(signal, sampling_rate=sampling_rate)
```

[18]:
```
RiskMarkers(Ste60_V3=0.04126003416910934, QTc=417, RA_V4=0.9481276953640756)
```

1.1.7 Perform differential diagnosis

[19]:
```
res = api.diagnose_with_risk_markers(signal, sampling_rate=sampling_rate)
print('Result:', res[0])
print('Explanation:')
display_text_explanation(res[1])
```

```
Result: Diagnosis.BER
Explanation:
Text explanation: Criterion value calculated as follows: (1.196 * [STE60 V3 in
mm]) + (0.059 * [QTc in ms]) - (0.326 * [RA V4 in mm])) = 22.00557372177566 did
not exceed the threshold 23.4, therefore the diagnosis is Benign Early
Repolarization
```

[20]:
```
res = api.check_BER_with_NN(signal)
print('Result:', res[0])
print('Explanation:')
display_text_and_image_explanation(res[1])
```

```
Load model at ./ECG/NN_based_approach/Models/Conv_ber_model.pt
Result: True
Explanation:
Text explanation: BER probability is 0.9278
GradCAM visualization:
```

[20]:

[21]:
```
res = api.check_MI_with_NN(signal)
print('Result:', res[0])
print('Explanation:')
display_text_and_image_explanation(res[1])
```

```
Load model at ./ECG/NN_based_approach/Models/Conv_mi_model.pt
Result: False
Explanation:
Text explanation: MI probability is 0.0017
GradCAM visualization:
```

[21]:

1.1.8 Check if the ECG is normal or not

[25]:
```
from ECG.ecghealthcheck.enums import ECGClass
```

[27]:
```
signal = signal[:, :4000]
signal.shape
```

[27]:
```
(12, 4000)
```

[29]:
```
res = api.check_ecg_is_normal(signal, ECGClass.ALL)
print('Result:', res[0])
display_text_explanation(res[1])
```

```
Result: True
Text explanation: The signal is ok
```