

Inference 2D

May 3, 2025

1 Importing Necessary Libraries

```
[2]: import random
import numpy as np
import tensorflow as tf
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
```

2 Loading the Model and the Data

```
[5]: model = tf.keras.models.load_model("../Models/Model 2D.h5")
```

```
[7]: X, y = np.load("../Features.npy"), np.load("../Labels (Mutli Class).npy")
```

2.1 Splitting the Data

```
[10]: _, X_test, _, y_test = train_test_split(X, y, test_size = 0.2, shuffle = True,
↳ random_state = 30)
```

```
[12]: X_test.shape, y_test.shape
```

```
[12]: ((21884, 250, 2), (21884,))
```

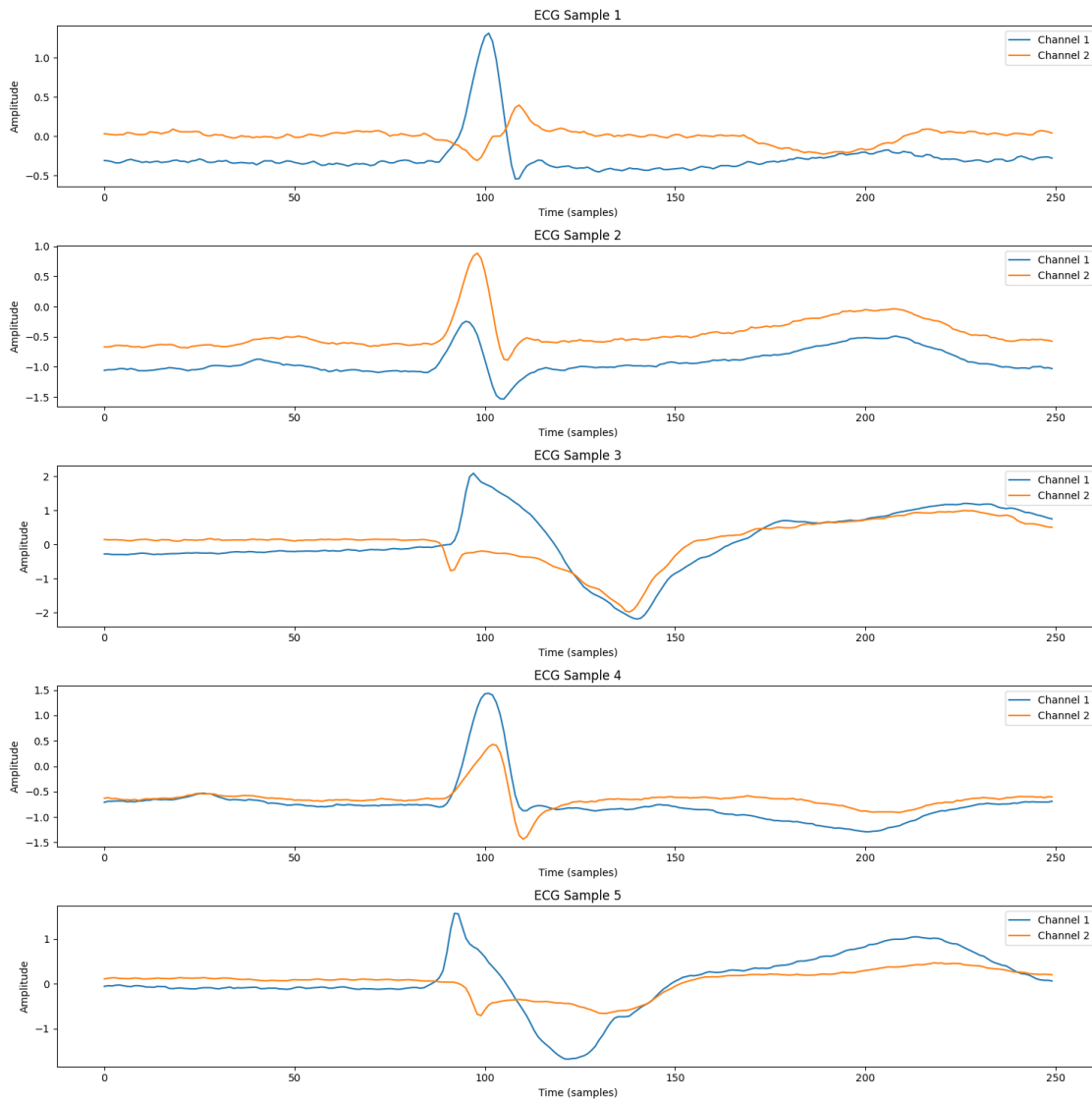
2.2 Plotting Random Samples

```
[15]: num_samples = 5
fig, axs = plt.subplots(num_samples, 1, figsize=(15, 15))

for i in range(num_samples):
    axs[i].plot(X_test[i, :, 0], label='Channel 1')
    axs[i].plot(X_test[i, :, 1], label='Channel 2')
    axs[i].set_title(f'ECG Sample {i + 1}')
    axs[i].set_xlabel('Time (samples)')
    axs[i].set_ylabel('Amplitude')
    axs[i].legend()

plt.tight_layout()
```

```
plt.show()
```



3 Inference

Abnormal Beat Symbols

```
abnormal = ['L', 'R', 'V', '/', 'A', 'f', 'F', 'j', 'a', 'E', 'J', 'e', 'S']
```

Normal Beat Symbols

```
normal = ['N']
```

```
[19]: symbol, code = [], []
```

```
with open("../Remapped_Symbol_Classes.txt", "r", encoding = "utf-8") as file:
```

```

for line in file:
    if "→" in line:
        s, c = line.strip().split("→")
        symbol.append(s.rstrip())
        code.append(c.lstrip())

```

```
[21]: symbol, code
```

```
[21]: ([ '/', 'A', 'E', 'F', 'J', 'L', 'N', 'R', 'S', 'V', 'a', 'e', 'f', 'j'],
      ['0', '1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12', '13'])
```

```
[33]: def infer(input):
      temp = np.expand_dims(input, axis = 0)
      idx = np.argmax(model.predict(temp))

      plt.figure(figsize = (10, 6))
      plt.plot(input)
      plt.title(f"Random ECG Sample | Prediction: {'Normal' if idx == 6 else 'Abnormal'} and Annotation: {symbol[idx]} | Actual: {symbol[y_test[i]]}")
      plt.grid()
      plt.xlabel("Time")
      plt.ylabel("Amplitude")
      plt.show()

```

```
[85]: i = random.randint(0, 21884)
      infer(X_test[i, :, :])
```

1/1 [=====] - 0s 16ms/step

