Dataprocessing+train+inference

December 13, 2023

[ ]:

```python
from DeepEEG import input_preparation
from DeepEEG.input_preparation import get_filepaths
from DeepEEG.input_preparation import get_labels
from DeepEEG.input_preparation import get_data
import pandas as pd
```

[ ]:

```python
root_folder = "Data"
training_filepaths = get_filepaths(root_folder)
labels = get_labels(root_folder)
training_filepaths, labels
```

[ ]:

```
({'Data/Speaking/s 1cleaned.csv': 'Speaking',
  'Data/Speaking/s 3cleaned.csv': 'Speaking',
  'Data/Speaking/s 4cleaned.csv': 'Speaking',
  'Data/Speaking/s 2cleaned.csv': 'Speaking',
  'Data/Reading/r 5cleaned.csv': 'Reading',
  'Data/Reading/r 3cleaned.csv': 'Reading',
  'Data/Reading/r 4cleaned.csv': 'Reading',
  'Data/Reading/r 6cleaned.csv': 'Reading',
  'Data/Reading/r 1cleaned.csv': 'Reading',
  'Data/Reading/r 2cleaned.csv': 'Reading',
  'Data/Watching/w 1cleaned.csv': 'Watching',
  'Data/Watching/w 5cleaned.csv': 'Watching',
  'Data/Watching/w 2cleaned.csv': 'Watching',
  'Data/Watching/w 4cleaned.csv': 'Watching',
  'Data/Watching/w 3cleaned.csv': 'Watching',
  'Data/Watching/w 6cleaned.csv': 'Watching'},
 {'Speaking': 0, 'Reading': 1, 'Watching': 2})
```

[ ]:

```python
def get_list(filepaths):
    list = []
    for i in filepaths.keys():
        list.append(i)
    return(list)
```

[ ]:

```python
list_data_dict = get_list(training_filepaths)
for i, data_dict in enumerate(list_data_dict):
    print(f"Index: {i}, Dictionary: {data_dict}")
```

```
Index: 0, Dictionary: Data/Speaking/s 1cleaned.csv
Index: 1, Dictionary: Data/Speaking/s 3cleaned.csv
Index: 2, Dictionary: Data/Speaking/s 4cleaned.csv
Index: 3, Dictionary: Data/Speaking/s 2cleaned.csv
Index: 4, Dictionary: Data/Reading/r 5cleaned.csv
Index: 5, Dictionary: Data/Reading/r 3cleaned.csv
Index: 6, Dictionary: Data/Reading/r 4cleaned.csv
Index: 7, Dictionary: Data/Reading/r 6cleaned.csv
Index: 8, Dictionary: Data/Reading/r 1cleaned.csv
Index: 9, Dictionary: Data/Reading/r 2cleaned.csv
Index: 10, Dictionary: Data/Watching/w 1cleaned.csv
Index: 11, Dictionary: Data/Watching/w 5cleaned.csv
Index: 12, Dictionary: Data/Watching/w 2cleaned.csv
Index: 13, Dictionary: Data/Watching/w 4cleaned.csv
Index: 14, Dictionary: Data/Watching/w 3cleaned.csv
Index: 15, Dictionary: Data/Watching/w 6cleaned.csv
```

[ ]:

```python
datasignals_s_1, one_hot_s_1, label_s_1 = get_data(
list_data_dict[0],
labels, training_filepaths)
datasignals_s_2, one_hot_s_2, label_s_2 = get_data(
list_data_dict[3],labels, training_filepaths)
datasignals_s_3, one_hot_s_3, label_s_3 = get_data(
list_data_dict[1],labels, training_filepaths)
datasignals_s_4, one_hot_s_4, label_s_4 = get_data(
list_data_dict[2],labels, training_filepaths)
datasignals_r_1, one_hot_r_1, label_r_1 = get_data(
list_data_dict[8],
labels, training_filepaths)

datasignals_r_2, one_hot_r_2, label_r_2 = get_data(
list_data_dict[9],labels, training_filepaths)
datasignals_r_3, one_hot_r_3, label_r_3 = get_data(
list_data_dict[5],labels, training_filepaths)
datasignals_r_4, one_hot_r_4, label_r_4 = get_data(
list_data_dict[6],labels, training_filepaths)
datasignals_r_5, one_hot_r_5, label_r_5 = get_data(
list_data_dict[4],labels, training_filepaths)
datasignals_r_6, one_hot_r_6, label_r_6 = get_data(
list_data_dict[7],labels, training_filepaths)



datasignals_w_1, one_hot_w_1, label_w_1 = get_data(
list_data_dict[10],labels, training_filepaths)
datasignals_w_2, one_hot_w_2, label_w_2 = get_data(
list_data_dict[12],labels, training_filepaths)

datasignals_w_3, one_hot_w_3, label_w_3 = get_data(
list_data_dict[14],labels, training_filepaths)

datasignals_w_4, one_hot_w_4, label_w_4 = get_data(
list_data_dict[13],labels, training_filepaths)

datasignals_w_5, one_hot_w_5, label_w_5 = get_data(
list_data_dict[11],labels, training_filepaths)

datasignals_w_6, one_hot_w_6, label_w_6 = get_data(
list_data_dict[15],labels, training_filepaths)
```

[ ]:

```
concatenated_data_s = pd.concat([datasignals_s_1,
datasignals_s_2,datasignals_s_3,datasignals_s_4], axis=0,
          ignore_index=True)

concatenated_data_r = pd.concat([ datasignals_r_1,
datasignals_r_2, datasignals_r_3,datasignals_r_4,
datasignals_r_5, datasignals_r_6], axis=0,
ignore_index=True)

concatenated_data_w = pd.concat([
    datasignals_w_1, datasignals_w_2, datasignals_w_3,
    datasignals_w_4, datasignals_w_5, datasignals_w_6
], axis=0,ignore_index=True)
```

[ ]:

```
all_Data_connected = pd.concat([
concatenated_data_s, concatenated_data_r, concatenated_data_w
])
```

[ ]:

```
import plotly.graph_objects as go
from plotly.subplots import make_subplots

# Assuming you have concatenated your data into concatenated_data DataFrame
# Extracting EEG1, EEG2, Acc_X, Acc_Y, Acc_Z data
eeg1_data = concatenated_data_s['EEG1']
eeg2_data = concatenated_data_s['EEG2']
acc_x_data = concatenated_data_s['Acc_X']
acc_y_data = concatenated_data_s['Acc_Y']
acc_z_data = concatenated_data_s['Acc_Z']

# Create subplots
fig = make_subplots(rows=5, cols=1, shared_xaxes=True, vertical_spacing=0.03)
```

```
# Add EEG1 trace to subplot 1
fig.add_trace(go.Scatter(x=concatenated_data_s.index, y=eeg1_data,␣
 ↪mode='lines', name='EEG1'), row=1, col=1)

# Add EEG2 trace to subplot 2
fig.add_trace(go.Scatter(x=concatenated_data_s.index, y=eeg2_data,␣
 ↪mode='lines', name='EEG2'), row=2, col=1)

# Add Acc_X trace to subplot 3
fig.add_trace(go.Scatter(x=concatenated_data_s.index, y=acc_x_data,␣
 ↪mode='lines', name='Acc_X'), row=3, col=1)

# Add Acc_Y trace to subplot 4
fig.add_trace(go.Scatter(x=concatenated_data_s.index, y=acc_y_data,␣
 ↪mode='lines', name='Acc_Y'), row=4, col=1)

# Add Acc_Z trace to subplot 5
fig.add_trace(go.Scatter(x=concatenated_data_s.index, y=acc_z_data,␣
 ↪mode='lines', name='Acc_Z'), row=5, col=1)

# Update layout
fig.update_layout(
    height=800,
    title_text="EEG and Accelerometer Signals speaking data",
    xaxis=dict(title='Index'),
    showlegend=True,
    legend=dict(
        orientation="h",
        yanchor="bottom",
        y=1.02,
        xanchor="right",
        x=1
    )
)
fig.show()
```

[ ]:

```
import plotly.graph_objects as go
from plotly.subplots import make_subplots

# Assuming you have concatenated your data into concatenated_data DataFrame
# Assuming the data is indexed by the sample number and contains EEG1, EEG2,␣
 ↪Acc_X, Acc_Y, Acc_Z data in microvolts (µV)

# Calculate time in seconds based on the sampling rate (220 Hz)
sampling_rate = 220  # Hz
```

```
time_seconds = concatenated_data_s.index / sampling_rate

# Extracting EEG1, EEG2, Acc_X, Acc_Y, Acc_Z data
eeg1_data = concatenated_data_s['EEG1']
eeg2_data = concatenated_data_s['EEG2']
acc_x_data = concatenated_data_s['Acc_X']
acc_y_data = concatenated_data_s['Acc_Y']
acc_z_data = concatenated_data_s['Acc_Z']

# Create subplots
fig = make_subplots(rows=5, cols=1, shared_xaxes=True, vertical_spacing=0.03)

# Add EEG1 trace to subplot 1
fig.add_trace(go.Scatter(x=time_seconds, y=eeg1_data, mode='lines',␣
 ↪name='EEG1'), row=1, col=1)

# Add EEG2 trace to subplot 2
fig.add_trace(go.Scatter(x=time_seconds, y=eeg2_data, mode='lines',␣
 ↪name='EEG2'), row=2, col=1)

# Add Acc_X trace to subplot 3
fig.add_trace(go.Scatter(x=time_seconds, y=acc_x_data, mode='lines',␣
 ↪name='Acc_X'), row=3, col=1)

# Add Acc_Y trace to subplot 4
fig.add_trace(go.Scatter(x=time_seconds, y=acc_y_data, mode='lines',␣
 ↪name='Acc_Y'), row=4, col=1)

# Add Acc_Z trace to subplot 5
fig.add_trace(go.Scatter(x=time_seconds, y=acc_z_data, mode='lines',␣
 ↪name='Acc_Z'), row=5, col=1)

# Update layout with legends and titles
fig.update_layout(
    height=800,
    title_text="EEG and Accelerometer Signals speaking data",
    xaxis=dict(title='Time (seconds)'),
    showlegend=True,
    legend=dict(
        orientation="h",
        yanchor="bottom",
        y=1.02,
        xanchor="right",
        x=1
    )
)
```

```
fig.show()
```

[ ]:

```
import plotly.graph_objects as go
from plotly.subplots import make_subplots

# Assuming you have concatenated your data into concatenated_data DataFrame
# Extracting EEG1, EEG2, Acc_X, Acc_Y, Acc_Z data
eeg1_data = all_Data_connected['EEG1']
eeg2_data = all_Data_connected['EEG2']
acc_x_data = all_Data_connected['Acc_X']
acc_y_data = all_Data_connected['Acc_Y']
acc_z_data = all_Data_connected['Acc_Z']

# Create subplots
fig = make_subplots(rows=5, cols=1, shared_xaxes=True, vertical_spacing=0.03)

# Add EEG1 trace to subplot 1
fig.add_trace(go.Scatter(x=all_Data_connected.index, y=eeg1_data, mode='lines',␣
 ↪name='EEG1'), row=1, col=1)

# Add EEG2 trace to subplot 2
fig.add_trace(go.Scatter(x=all_Data_connected.index, y=eeg2_data, mode='lines',␣
 ↪name='EEG2'), row=2, col=1)

# Add Acc_X trace to subplot 3
fig.add_trace(go.Scatter(x=all_Data_connected.index, y=acc_x_data,␣
 ↪mode='lines', name='Acc_X'), row=3, col=1)

# Add Acc_Y trace to subplot 4
fig.add_trace(go.Scatter(x=all_Data_connected.index, y=acc_y_data,␣
 ↪mode='lines', name='Acc_Y'), row=4, col=1)

# Add Acc_Z trace to subplot 5
fig.add_trace(go.Scatter(x=all_Data_connected.index, y=acc_z_data,␣
 ↪mode='lines', name='Acc_Z'), row=5, col=1)

# Update layout with legends and titles
fig.update_layout(
    height=800,
    title_text="EEG and Accelerometer Signals all Data",
    xaxis=dict(title='Index'),
    showlegend=True,
    legend=dict(
        orientation="h",
        yanchor="bottom",
        y=1.02,
        xanchor="right",
        x=1
    )
)
fig.show()
```

[ ]:

```
import plotly.graph_objects as go
from plotly.subplots import make_subplots

# Assuming you have concatenated
# your data into concatenated_data DataFrame

# Calculate time in seconds based on the sampling rate (220 Hz)
sampling_rate = 220  # Hz
time_seconds = all_Data_connected.index / sampling_rate

# Extracting EEG1, EEG2, Acc_X, Acc_Y, Acc_Z data
eeg1_data = all_Data_connected['EEG1']
eeg2_data = all_Data_connected['EEG2']
acc_x_data = all_Data_connected['Acc_X']
acc_y_data = all_Data_connected['Acc_Y']
acc_z_data = all_Data_connected['Acc_Z']

# Create subplots
fig = make_subplots(rows=5, cols=1,
     shared_xaxes=True, vertical_spacing=0.03)

# Add EEG1 trace to subplot 1
fig.add_trace(go.Scatter(x=time_seconds,
    y=eeg1_data, mode='lines', name='EEG1'), row=1, col=1)

# Add EEG2 trace to subplot 2
fig.add_trace(go.Scatter(x=time_seconds,
y=eeg2_data, mode='lines', name='EEG2'), row=2, col=1)

# Add Acc_X trace to subplot 3
fig.add_trace(go.Scatter(x=time_seconds,
y=acc_x_data, mode='lines', name='Acc_X'), row=3, col=1)

# Add Acc_Y trace to subplot 4
fig.add_trace(go.Scatter(x=time_seconds,
y=acc_y_data, mode='lines', name='Acc_Y'), row=4, col=1)

# Add Acc_Z trace to subplot 5
fig.add_trace(go.Scatter(x=time_seconds,
y=acc_z_data, mode='lines', name='Acc_Z'), row=5, col=1)
```

```
# Update layout with legends and titles
fig.update_layout(
    height=800,
    title_text="EEG and Accelerometer Signals all",
    xaxis=dict(title='Time (seconds)'),
    showlegend=True,
    legend=dict(
        orientation="h",
        yanchor="bottom",
        y=1.02,
        xanchor="right",
        x=1
    )
)


fig.show()
```

[ ]:

```
all_Data_connected
```

[ ]:

```
             EEG1        EEG2       Acc_X       Acc_Y       Acc_Z
0      866.904663  842.229919 -531.250854  859.376343  238.281616
1      845.519897  822.490173 -519.532043  855.470093  246.094132
2      860.324707  835.650024 -519.532043  855.470093  246.094132
3      860.324707  835.650024 -542.969604  851.563843  238.281616
4      835.650024  837.294983 -535.157104  859.376343  238.281616
…             …          …           …          …           …
31642  847.164856  857.034729 -703.126099  750.001160  136.718964
31643  875.129517  837.294983 -703.126099  750.001160  136.718964
31644  852.099793  837.294983 -703.126099  750.001160  136.718964
31645  832.360046  870.194580 -707.032349  746.094910  132.812714
31646  843.874939  843.874939 -703.126099  746.094910  136.718964

[104480 rows x 5 columns]
```

[ ]: