AAI521_Final_Jeremy_V1

December 11, 2023

[ ]:
```
!pip install numpy pandas matplotlib opencv-python tensorflow
```

```
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages
(1.23.5)
Requirement already satisfied: pandas in /usr/local/lib/python3.10/dist-packages
(1.5.3)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dist-
packages (3.7.1)
Requirement already satisfied: opencv-python in /usr/local/lib/python3.10/dist-
packages (4.8.0.76)
Requirement already satisfied: tensorflow in /usr/local/lib/python3.10/dist-
packages (2.14.0)
Requirement already satisfied: python-dateutil>=2.8.1 in
/usr/local/lib/python3.10/dist-packages (from pandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-
packages (from pandas) (2023.3.post1)
Requirement already satisfied: contourpy>=1.0.1 in
/usr/local/lib/python3.10/dist-packages (from matplotlib) (1.2.0)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-
packages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in
/usr/local/lib/python3.10/dist-packages (from matplotlib) (4.44.3)
Requirement already satisfied: kiwisolver>=1.0.1 in
/usr/local/lib/python3.10/dist-packages (from matplotlib) (1.4.5)
Requirement already satisfied: packaging>=20.0 in
/usr/local/lib/python3.10/dist-packages (from matplotlib) (23.2)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/dist-
packages (from matplotlib) (9.4.0)
Requirement already satisfied: pyparsing>=2.3.1 in
/usr/local/lib/python3.10/dist-packages (from matplotlib) (3.1.1)
Requirement already satisfied: absl-py>=1.0.0 in /usr/local/lib/python3.10/dist-
packages (from tensorflow) (1.4.0)
Requirement already satisfied: astunparse>=1.6.0 in
/usr/local/lib/python3.10/dist-packages (from tensorflow) (1.6.3)
Requirement already satisfied: flatbuffers>=23.5.26 in
/usr/local/lib/python3.10/dist-packages (from tensorflow) (23.5.26)
Requirement already satisfied: gast!=0.5.0,!=0.5.1,!=0.5.2,>=0.2.1 in
/usr/local/lib/python3.10/dist-packages (from tensorflow) (0.5.4)
```

```
Requirement already satisfied: google-pasta>=0.1.1 in
/usr/local/lib/python3.10/dist-packages (from tensorflow) (0.2.0)
Requirement already satisfied: h5py>=2.9.0 in /usr/local/lib/python3.10/dist-
packages (from tensorflow) (3.9.0)
Requirement already satisfied: libclang>=13.0.0 in
/usr/local/lib/python3.10/dist-packages (from tensorflow) (16.0.6)
Requirement already satisfied: ml-dtypes==0.2.0 in
/usr/local/lib/python3.10/dist-packages (from tensorflow) (0.2.0)
Requirement already satisfied: opt-einsum>=2.3.2 in
/usr/local/lib/python3.10/dist-packages (from tensorflow) (3.3.0)
Requirement already satisfied:
protobuf!=4.21.0,!=4.21.1,!=4.21.2,!=4.21.3,!=4.21.4,!=4.21.5,<5.0.0dev,>=3.20.3
in /usr/local/lib/python3.10/dist-packages (from tensorflow) (3.20.3)
Requirement already satisfied: setuptools in /usr/local/lib/python3.10/dist-
packages (from tensorflow) (67.7.2)
Requirement already satisfied: six>=1.12.0 in /usr/local/lib/python3.10/dist-
packages (from tensorflow) (1.16.0)
Requirement already satisfied: termcolor>=1.1.0 in
/usr/local/lib/python3.10/dist-packages (from tensorflow) (2.3.0)
Requirement already satisfied: typing-extensions>=3.6.6 in
/usr/local/lib/python3.10/dist-packages (from tensorflow) (4.5.0)
Requirement already satisfied: wrapt<1.15,>=1.11.0 in
/usr/local/lib/python3.10/dist-packages (from tensorflow) (1.14.1)
Requirement already satisfied: tensorflow-io-gcs-filesystem>=0.23.1 in
/usr/local/lib/python3.10/dist-packages (from tensorflow) (0.34.0)
Requirement already satisfied: grpcio<2.0,>=1.24.3 in
/usr/local/lib/python3.10/dist-packages (from tensorflow) (1.59.2)
Requirement already satisfied: tensorboard<2.15,>=2.14 in
/usr/local/lib/python3.10/dist-packages (from tensorflow) (2.14.1)
Requirement already satisfied: tensorflow-estimator<2.15,>=2.14.0 in
/usr/local/lib/python3.10/dist-packages (from tensorflow) (2.14.0)
Requirement already satisfied: keras<2.15,>=2.14.0 in
/usr/local/lib/python3.10/dist-packages (from tensorflow) (2.14.0)
Requirement already satisfied: wheel<1.0,>=0.23.0 in
/usr/local/lib/python3.10/dist-packages (from astunparse>=1.6.0->tensorflow)
(0.41.3)
Requirement already satisfied: google-auth<3,>=1.6.3 in
/usr/local/lib/python3.10/dist-packages (from
tensorboard<2.15,>=2.14->tensorflow) (2.17.3)
Requirement already satisfied: google-auth-oauthlib<1.1,>=0.5 in
/usr/local/lib/python3.10/dist-packages (from
tensorboard<2.15,>=2.14->tensorflow) (1.0.0)
Requirement already satisfied: markdown>=2.6.8 in
/usr/local/lib/python3.10/dist-packages (from
tensorboard<2.15,>=2.14->tensorflow) (3.5.1)
Requirement already satisfied: requests<3,>=2.21.0 in
/usr/local/lib/python3.10/dist-packages (from
tensorboard<2.15,>=2.14->tensorflow) (2.31.0)
```

```
Requirement already satisfied: tensorboard-data-server<0.8.0,>=0.7.0 in
/usr/local/lib/python3.10/dist-packages (from
tensorboard<2.15,>=2.14->tensorflow) (0.7.2)
Requirement already satisfied: werkzeug>=1.0.1 in
/usr/local/lib/python3.10/dist-packages (from
tensorboard<2.15,>=2.14->tensorflow) (3.0.1)
Requirement already satisfied: cachetools<6.0,>=2.0.0 in
/usr/local/lib/python3.10/dist-packages (from google-
auth<3,>=1.6.3->tensorboard<2.15,>=2.14->tensorflow) (5.3.2)
Requirement already satisfied: pyasn1-modules>=0.2.1 in
/usr/local/lib/python3.10/dist-packages (from google-
auth<3,>=1.6.3->tensorboard<2.15,>=2.14->tensorflow) (0.3.0)
Requirement already satisfied: rsa<5,>=3.1.4 in /usr/local/lib/python3.10/dist-
packages (from google-auth<3,>=1.6.3->tensorboard<2.15,>=2.14->tensorflow) (4.9)
Requirement already satisfied: requests-oauthlib>=0.7.0 in
/usr/local/lib/python3.10/dist-packages (from google-auth-
oauthlib<1.1,>=0.5->tensorboard<2.15,>=2.14->tensorflow) (1.3.1)
Requirement already satisfied: charset-normalizer<4,>=2 in
/usr/local/lib/python3.10/dist-packages (from
requests<3,>=2.21.0->tensorboard<2.15,>=2.14->tensorflow) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-
packages (from requests<3,>=2.21.0->tensorboard<2.15,>=2.14->tensorflow) (3.4)
Requirement already satisfied: urllib3<3,>=1.21.1 in
/usr/local/lib/python3.10/dist-packages (from
requests<3,>=2.21.0->tensorboard<2.15,>=2.14->tensorflow) (2.0.7)
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.10/dist-packages (from
requests<3,>=2.21.0->tensorboard<2.15,>=2.14->tensorflow) (2023.7.22)
Requirement already satisfied: MarkupSafe>=2.1.1 in
/usr/local/lib/python3.10/dist-packages (from
werkzeug>=1.0.1->tensorboard<2.15,>=2.14->tensorflow) (2.1.3)
Requirement already satisfied: pyasn1<0.6.0,>=0.4.6 in
/usr/local/lib/python3.10/dist-packages (from pyasn1-modules>=0.2.1->google-
auth<3,>=1.6.3->tensorboard<2.15,>=2.14->tensorflow) (0.5.0)
Requirement already satisfied: oauthlib>=3.0.0 in
/usr/local/lib/python3.10/dist-packages (from requests-oauthlib>=0.7.0->google-
auth-oauthlib<1.1,>=0.5->tensorboard<2.15,>=2.14->tensorflow) (3.2.2)
```

[ ]:
```
# Install the Kaggle API
!pip install kaggle
```

```
Requirement already satisfied: kaggle in /usr/local/lib/python3.10/dist-packages
(1.5.16)
Requirement already satisfied: six>=1.10 in /usr/local/lib/python3.10/dist-
packages (from kaggle) (1.16.0)
Requirement already satisfied: certifi in /usr/local/lib/python3.10/dist-
packages (from kaggle) (2023.7.22)
Requirement already satisfied: python-dateutil in
```

```
/usr/local/lib/python3.10/dist-packages (from kaggle) (2.8.2)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-
packages (from kaggle) (2.31.0)
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages
(from kaggle) (4.66.1)
Requirement already satisfied: python-slugify in /usr/local/lib/python3.10/dist-
packages (from kaggle) (8.0.1)
Requirement already satisfied: urllib3 in /usr/local/lib/python3.10/dist-
packages (from kaggle) (2.0.7)
Requirement already satisfied: bleach in /usr/local/lib/python3.10/dist-packages
(from kaggle) (6.1.0)
Requirement already satisfied: webencodings in /usr/local/lib/python3.10/dist-
packages (from bleach->kaggle) (0.5.1)
Requirement already satisfied: text-unidecode>=1.3 in
/usr/local/lib/python3.10/dist-packages (from python-slugify->kaggle) (1.3)
Requirement already satisfied: charset-normalizer<4,>=2 in
/usr/local/lib/python3.10/dist-packages (from requests->kaggle) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-
packages (from requests->kaggle) (3.4)
```

[ ]:
```
# Upload the Kaggle API token
from google.colab import files
uploaded = files.upload()
```

```
<IPython.core.display.HTML object>

Saving kaggle.json to kaggle.json
```

[ ]:
```
# Move the uploaded file to the required directory
!mkdir -p ~/.kaggle
!mv kaggle.json ~/.kaggle/
!chmod 600 ~/.kaggle/kaggle.json
```

[ ]:
```
# Download the ASL dataset from Kaggle
!kaggle competitions download -c asl-signs
```

```
Downloading asl-signs.zip to /content
100% 37.3G/37.4G [05:44<00:00, 115MB/s]
100% 37.4G/37.4G [05:44<00:00, 116MB/s]
```

[ ]:
```
# Unzip the downloaded dataset
!unzip -q asl-signs.zip
```

[ ]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import cv2
from tqdm.notebook import tqdm
import tensorflow as tf
```

```
import json
import os
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, LSTM
from tensorflow.keras.utils import to_categorical
```

[ ]:
```
# Read the CSV file
df = pd.read_csv('train.csv')

# Display basic information about the dataset
df
```

[ ]:
```
                                                  path  participant_id  \
0      train_landmark_files/26734/1000035562.parquet           26734
1      train_landmark_files/28656/1000106739.parquet           28656
2       train_landmark_files/16069/100015657.parquet           16069
3      train_landmark_files/25571/1000210073.parquet           25571
4      train_landmark_files/62590/1000240708.parquet           62590
…                                                ...             …
94472   train_landmark_files/53618/999786174.parquet           53618
94473   train_landmark_files/26734/999799849.parquet           26734
94474   train_landmark_files/25571/999833418.parquet           25571
94475   train_landmark_files/29302/999895257.parquet           29302
94476   train_landmark_files/36257/999962374.parquet           36257

       sequence_id    sign
0       1000035562    blow
1       1000106739    wait
2        100015657   cloud
3       1000210073    bird
4       1000240708    owie
…              …     …
94472    999786174   white
94473    999799849    have
94474    999833418  flower
94475    999895257    room
94476    999962374   happy

[94477 rows x 4 columns]
```

[ ]:
```
# Check for missing values
print(df.isnull().sum())
```

```
path              0
participant_id    0
sequence_id       0
```

```
sign              0
dtype: int64
```

[ ]:
```
# Load the sign index mapping from the JSON file
with open('sign_to_prediction_index_map.json', 'r') as f:
    sign_index_mapping = json.load(f)
```

[ ]:
```
# Convert the sign index mapping to a DataFrame
sign_index_df = pd.DataFrame(list(sign_index_mapping.items()), columns=['sign',␣
 ↪'sign_info'])

# Display basic information about the sign index DataFrame
sign_index_df.head()
```

[ ]:
```
        sign  sign_info
0         TV          0
1      after          1
2   airplane          2
3        all          3
4  alligator          4
```

[ ]:
```
# Merge the two DataFrames based on the 'sign' column
merged_df = pd.merge(df, sign_index_df, how='left', on='sign')

# Display basic information about the merged DataFrame
merged_df.head()
```

[ ]:
```
                                        path  participant_id  sequence_id  \
0  train_landmark_files/26734/1000035562.parquet           26734   1000035562
1  train_landmark_files/28656/1000106739.parquet           28656   1000106739
2   train_landmark_files/16069/100015657.parquet           16069    100015657
3  train_landmark_files/25571/1000210073.parquet           25571   1000210073
4  train_landmark_files/62590/1000240708.parquet           62590   1000240708

    sign  sign_info
0   blow         25
1   wait        232
2  cloud         48
3   bird         23
4   owie        164
```

[ ]:
```
sample_fillede\['type'].unique()
```

[ ]:
```
array(['face', 'left_hand', 'pose', 'right_hand'], dtype=object)
```

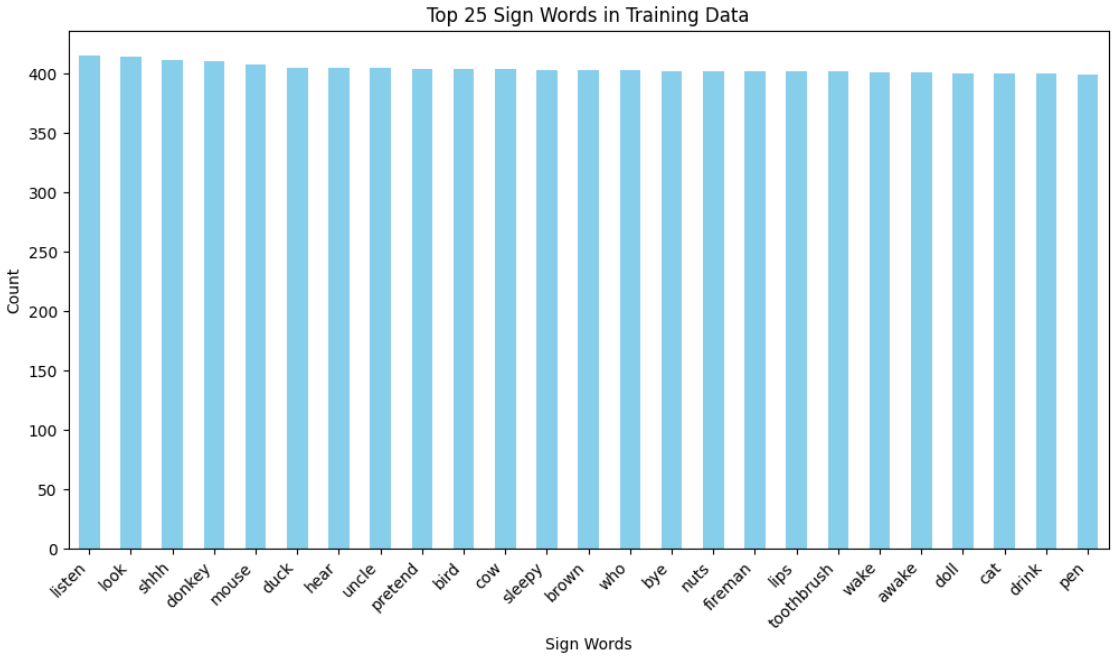
[ ]:
```
sample['frame'].unique()
```

[ ]:
```
array([17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28], dtype=int16)
```

[ ]:
```
# Assuming your DataFrame is named 'train_df'
top_signs = df['sign'].value_counts().head(25)

# Display the top 25 sign words
print(top_signs)
```

```
listen        415
look          414
shhh          411
donkey        410
mouse         408
duck          405
hear          405
uncle         405
pretend       404
bird          404
cow           404
sleepy        403
brown         403
who           403
bye           402
nuts          402
fireman       402
lips          402
toothbrush    402
wake          401
awake         401
doll          400
cat           400
drink         400
pen           399
Name: sign, dtype: int64
```

[ ]:
```
import matplotlib.pyplot as plt

# Plot the top 25 sign words
plt.figure(figsize=(12, 6))
top_signs.plot(kind='bar', color='skyblue')
plt.title('Top 25 Sign Words in Training Data')
plt.xlabel('Sign Words')
plt.ylabel('Count')
plt.xticks(rotation=45, ha='right')
plt.show()
```



[ ]:
```
#Read a Parquet file and set a sample
file_path = '/content/train_landmark_files/25571/1000210073.parquet'
landmark_sample = pd.read_parquet(file_path)

# Display the loaded data
landmark_sample
```

[ ]:

	frame	row_id	type	landmark_index	x	y	z
0	17	17-face-0	face	0	0.495870	0.478694	-0.037412
1	17	17-face-1	face	1	0.492222	0.447209	-0.067939
2	17	17-face-2	face	2	0.492067	0.457237	-0.035722
3	17	17-face-3	face	3	0.480419	0.415996	-0.050779
4	17	17-face-4	face	4	0.492035	0.437453	
…	…	…	…	…	…	…	
6511	28	28-right_hand-16	right_hand	16	0.506396	0.868416	
6512	28	28-right_hand-17	right_hand	17	0.323227	0.835990	
6513	28	28-right_hand-18	right_hand	18	0.435733	0.848917	
6514	28	28-right_hand-19	right_hand	19	0.476093	0.867098	
6515	28	28-right_hand-20	right_hand	20	0.488775	0.885244	

```
4    -0.072314
…         …
6511 -0.139545
6512 -0.136632
6513 -0.156200
6514 -0.149442
6515 -0.142629

[6516 rows x 7 columns]
```

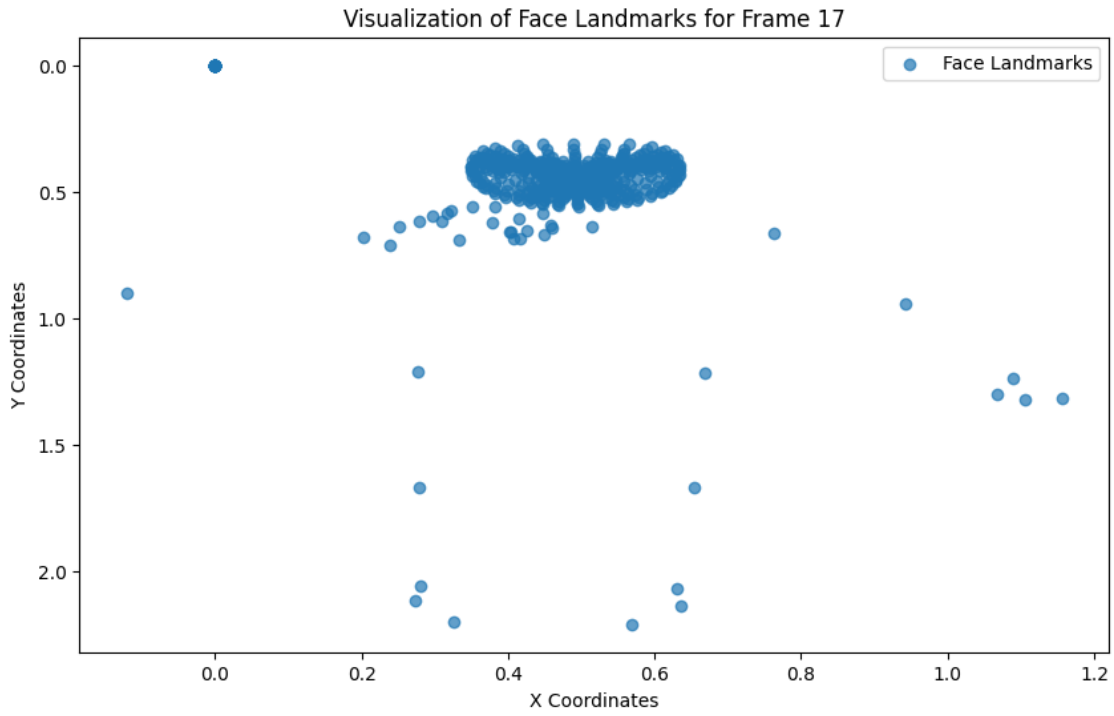
[ ]:
```
# Replace all NaN values with 0
sample = landmark_sample.fillna(0)

# Display the DataFrame with null values replaced
sample
```

[ ]:
```
      frame            row_id        type  landmark_index         x         y  \
0        17         17-face-0        face               0  0.495870  0.478694
1        17         17-face-1        face               1  0.492222  0.447209
2        17         17-face-2        face               2  0.492067  0.457237
3        17         17-face-3        face               3  0.480419  0.415996
4        17         17-face-4        face               4  0.492035  0.437453
…     …                 …         …                …        …       …
6511     28  28-right_hand-16  right_hand              16  0.506396  0.868416
6512     28  28-right_hand-17  right_hand              17  0.323227  0.835990
6513     28  28-right_hand-18  right_hand              18  0.435733  0.848917
6514     28  28-right_hand-19  right_hand              19  0.476093  0.867098
6515     28  28-right_hand-20  right_hand              20  0.488775  0.885244

             z
0    -0.037412
1    -0.067939
2    -0.035722
3    -0.050779
4    -0.072314
…         …
6511 -0.139545
6512 -0.136632
6513 -0.156200
6514 -0.149442
6515 -0.142629

[6516 rows x 7 columns]
```

[ ]:
```
# Filter the DataFrame for a specific frame
frame_103_face = sample[(sample['frame'] == 17)]
```

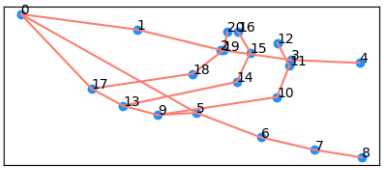
```
# Plot each type of landmark separately
plt.figure(figsize=(12, 8))

# Scatter plot for 'face' landmarks with reversed y-axis
plt.figure(figsize=(10, 6))
plt.scatter(frame_103_face['x'], frame_103_face['y'], label='Face Landmarks',␣
 ↪alpha=0.7)

# Reverse the y-axis
plt.gca().invert_yaxis()

# Set plot properties
plt.title('Visualization of Face Landmarks for Frame 17')
plt.xlabel('X Coordinates')
plt.ylabel('Y Coordinates')
plt.legend()
plt.show()
```

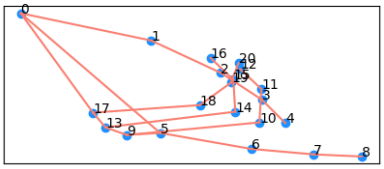
```
<Figure size 1200x800 with 0 Axes>
```

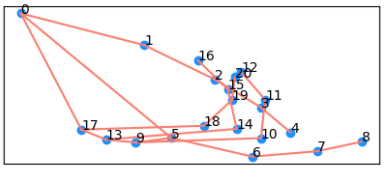


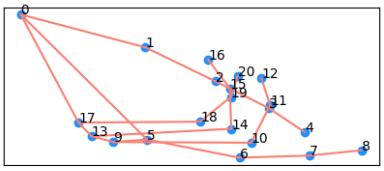
[ ]:
```
# pick the left hand and right hand points
sample_left_hand = sample[sample.type == "left_hand"]
sample_right_hand = sample[sample.type == "right_hand"]
```

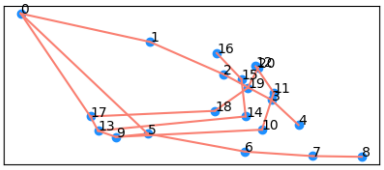
```
# display(sample_left_hand)

# edges that represents the hand edges
# How he knows the edges, so a mystery
edges =␣
 ↪[(0,1),(1,2),(2,3),(3,4),(0,5),(0,17),(5,6),(6,7),(7,8),(5,9),(9,10),(10,11),(11,12),
        (9,13),(13,14),(14,15),(15,16),(13,17),(17,18),(18,19),(19,20)]

# plotting a single frame into matplotlib
def plot_frame(df, frame_id, ax):
    df = df[df.frame == frame_id].sort_values(['landmark_index'])
    x = list(df.x)
    y = list(df.y)

    # plotting the points
    ax.scatter(df.x, df.y, color='dodgerblue')
    for i in range(len(x)):
        ax.text(x[i], y[i], str(i))

    # plotting the edges that represents the hand
    for edge in edges:
        ax.plot([x[edge[0]], x[edge[1]]], [y[edge[0]], y[edge[1]]],␣
 ↪color='salmon')
        ax.set_xlabel(f"Frame no. {frame_id}")
        ax.set_xticks([])
        ax.set_yticks([])
        ax.set_xticklabels([])
        ax.set_yticklabels([])

# plotting the multiple frames
def plot_frame_seq(df, frame_range, n_frames):
    frames = np.linspace(frame_range[0],frame_range[1],n_frames, dtype = int,␣
 ↪endpoint=True)
    fig, ax = plt.subplots(n_frames, 1, figsize=(5,25))
    for i in range(n_frames):
        plot_frame(df, frames[i], ax[i])

    plt.show()

plot_frame_seq(sample_right_hand, (17,28), 10)
```

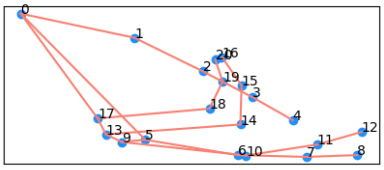

Frame no. 17

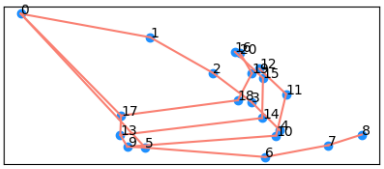

Frame no. 18

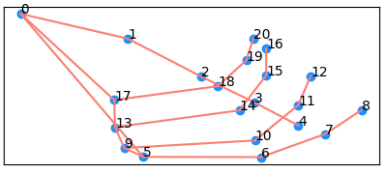

Frame no. 19

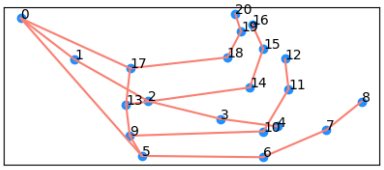

Frame no. 20

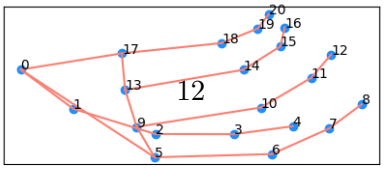

Frame no. 21


Frame no. 23


Frame no. 24


Frame no. 25


Frame no. 26


Frame no. 28

[ ]:
```
# Load landmark data (assuming you have a function to load Parquet files)
landmark_df = load_landmark_data('path/to/landmark_data.parquet')
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-49-53b7f909f2e5> in <cell line: 1>()
----> 1 merged_df1 = pd.merge(df, sample, on='participant_id', how='inner')

/usr/local/lib/python3.10/dist-packages/pandas/core/reshape/merge.py in␣
 ↪merge(left, right, how, on, left_on, right_on, left_index, right_index, sort,␣
 ↪suffixes, copy, indicator, validate)
    108     validate: str | None = None,
    109 ) -> DataFrame:
--> 110     op = _MergeOperation(
    111         left,
    112         right,

/usr/local/lib/python3.10/dist-packages/pandas/core/reshape/merge.py in␣
 ↪__init__(self, left, right, how, on, left_on, right_on, axis, left_index,␣
 ↪right_index, sort, suffixes, indicator, validate)
    701             self.right_join_keys,
    702             self.join_names,
--> 703         ) = self._get_merge_keys()
    704
    705         # validate the merge keys dtypes. We may need to coerce

/usr/local/lib/python3.10/dist-packages/pandas/core/reshape/merge.py in␣
 ↪_get_merge_keys(self)
   1160                         rk = cast(Hashable, rk)
   1161                         if rk is not None:
-> 1162                             right_keys.append(right.
 ↪_get_label_or_level_values(rk))
   1163                         else:
   1164                             # work-around for␣
 ↪merge_asof(right_index=True)

/usr/local/lib/python3.10/dist-packages/pandas/core/generic.py in␣
 ↪_get_label_or_level_values(self, key, axis)
   1848             )
   1849         else:
-> 1850             raise KeyError(key)
   1851
   1852         # Check for duplicates
```

```
KeyError: 'participant_id'
```

[ ]:
```python
# Function to load landmark data from Parquet files in a folder
def load_landmark_data(folder_path):
    combined_meta = {}
    for root, dirs, files in os.walk(folder_path):
        for file_name in tqdm(files):
            if file_name.endswith(".parquet"):
                file_path = os.path.join(root, file_name)
                example_landmark = pd.read_parquet(file_path)

                # Replace null values with 0
                example_landmark.fillna(0, inplace=True)

                # Get the number of landmarks with x, y, z data per type
                meta = example_landmark.dropna(subset=["x", "y", "z"])["type"].
 ↪value_counts().to_dict()
                meta["frames"] = example_landmark["frame"].nunique()

                # Calculate additional statistics if needed
                xyz_meta = (
                    example_landmark.agg(
                        {
                            "x": ["min", "max", "mean"],
                            "y": ["min", "max", "mean"],
                            "z": ["min", "max", "mean"],
                        }
                    )
                    .unstack()
                    .to_dict()
                )

                for key in xyz_meta.keys():
                    new_key = key[0] + "_" + key[1]
                    meta[new_key] = xyz_meta[key]

                combined_meta[file_path] = meta

    return combined_meta

# Specify the path to the root folder containing participant folders
root_folder_path = '/content/train_landmark_files'

# Load landmark data from all participant folders
combined_meta_all = {}
for participant_folder in tqdm(os.listdir(root_folder_path)):
```

```
    participant_folder_path = os.path.join(root_folder_path, participant_folder)

    if os.path.isdir(participant_folder_path):
        participant_combined_meta = load_landmark_data(participant_folder_path)
        combined_meta_all.update(participant_combined_meta)

# Create a DataFrame from the combined metadata
metadata_df = pd.DataFrame.from_dict(combined_meta_all, orient='index').
 ↪reset_index()
metadata_df.rename(columns={'index': 'file_path'}, inplace=True)

# Display the resulting DataFrame
print(metadata_df.head())
```

```
  0%|          | 0/21 [00:00<?, ?it/s]
  0%|          | 0/4826 [00:00<?, ?it/s]
  0%|          | 0/4841 [00:00<?, ?it/s]
  0%|          | 0/4753 [00:00<?, ?it/s]
  0%|          | 0/4677 [00:00<?, ?it/s]
  0%|          | 0/4810 [00:00<?, ?it/s]
  0%|          | 0/4563 [00:00<?, ?it/s]
  0%|          | 0/3499 [00:00<?, ?it/s]
  0%|          | 0/4968 [00:00<?, ?it/s]
  0%|          | 0/4900 [00:00<?, ?it/s]
  0%|          | 0/3865 [00:00<?, ?it/s]
  0%|          | 0/4545 [00:00<?, ?it/s]
  0%|          | 0/3502 [00:00<?, ?it/s]
  0%|          | 0/4722 [00:00<?, ?it/s]
  0%|          | 0/4563 [00:00<?, ?it/s]
  0%|          | 0/4782 [00:00<?, ?it/s]
  0%|          | 0/3338 [00:00<?, ?it/s]
  0%|          | 0/4656 [00:00<?, ?it/s]
  0%|          | 0/4848 [00:00<?, ?it/s]
  0%|          | 0/4648 [00:00<?, ?it/s]
  0%|          | 0/4896 [00:00<?, ?it/s]
  0%|          | 0/4275 [00:00<?, ?it/s]
```

```
                                                file_path   face  pose  left_hand  \
0  /content/train_landmark_files/55372/2802786652…   7956   561         357
1  /content/train_landmark_files/55372/3403106688…  14508  1023         651
2  /content/train_landmark_files/55372/1127624485…   8424   594         378
3  /content/train_landmark_files/55372/1559766834…   8424   594         378
4  /content/train_landmark_files/55372/657631983…   6552   462         294

   right_hand  frames     x_min     x_max    x_mean  y_min     y_max  \
0         357      17 -0.087367  1.199376  0.448886    0.0  2.479705
1         651      31 -0.240969  1.178582  0.419019    0.0  2.441859
2         378      18 -0.146753  1.073904  0.404705    0.0  2.518284
3         378      18 -0.069765  1.265994  0.388586    0.0  2.612595
4         294      14 -0.423106  1.303239  0.403132    0.0  2.532954

     y_mean     z_min     z_max    z_mean
0  0.386801 -3.059139  3.362435 -0.055478
1  0.370088 -2.872532  1.589201 -0.058870
2  0.403468 -2.520643  1.895188 -0.038549
3  0.378294 -2.927297  2.471197 -0.022748
4  0.364148 -2.680002  2.279785 -0.042742
```

[ ]:
```
metadata_df
```

[ ]:
```
                                                file_path   face  pose  \
0      /content/train_landmark_files/55372/2802786652…   7956   561
1      /content/train_landmark_files/55372/3403106688…  14508  1023
2      /content/train_landmark_files/55372/1127624485…   8424   594
3      /content/train_landmark_files/55372/1559766834…   8424   594
4      /content/train_landmark_files/55372/657631983…   6552   462
…                                                    …     …    …
94472  /content/train_landmark_files/27610/1696867677…  54756  3861
94473  /content/train_landmark_files/27610/2975578577…  49608  3498
94474  /content/train_landmark_files/27610/4223702977…  37440  2640
94475  /content/train_landmark_files/27610/558510995…   4680   330
94476  /content/train_landmark_files/27610/314634651…  11232   792

       left_hand  right_hand  frames     x_min     x_max    x_mean  y_min  \
0            357         357      17 -0.087367  1.199376  0.448886    0.0
1            651         651      31 -0.240969  1.178582  0.419019    0.0
2            378         378      18 -0.146753  1.073904  0.404705    0.0
3            378         378      18 -0.069765  1.265994  0.388586    0.0
4            294         294      14 -0.423106  1.303239  0.403132    0.0
…          …         …     …       …       …       …    …
94472       2457        2457     117 -0.117516  0.951807  0.261810    0.0
94473       2226        2226     106 -0.147046  0.976050  0.365789    0.0
94474       1680        1680      80 -0.079137  1.134678  0.466686    0.0
94475        210         210      10 -0.066891  0.939062  0.431534    0.0
```

```
94476        504         504      24 -0.053534  0.954912  0.390166    0.0

          y_max    y_mean     z_min     z_max    z_mean
0      2.479705  0.386801 -3.059139  3.362435 -0.055478
1      2.441859  0.370088 -2.872532  1.589201 -0.058870
2      2.518284  0.403468 -2.520643  1.895188 -0.038549
3      2.612595  0.378294 -2.927297  2.471197 -0.022748
4      2.532954  0.364148 -2.680002  2.279785 -0.042742
…           …         …         …         …         …
94472  2.467156  0.463274 -2.706611  1.537550 -0.037171
94473  2.550603  0.493334 -2.661751  1.042857 -0.050042
94474  2.357260  0.496470 -2.643354  1.984523 -0.032431
94475  2.559409  0.609520 -2.620925  1.503764 -0.050653
94476  2.431209  0.508731 -2.105850  1.693273 -0.027899

[94477 rows x 15 columns]
```

[ ]:
```
# Assuming your DataFrame is named 'df'
metadata_df['file_path'] = metadata_df['file_path'].str.replace('/content/', '')

# Display the updated DataFrame
metadata_df
```

[ ]:
```
                                          file_path   face  pose  left_hand  \
0      train_landmark_files/55372/2802786652.parquet   7956   561        357
1      train_landmark_files/55372/3403106688.parquet  14508  1023        651
2      train_landmark_files/55372/1127624485.parquet   8424   594        378
3      train_landmark_files/55372/1559766834.parquet   8424   594        378
4       train_landmark_files/55372/657631983.parquet   6552   462        294
…                                                 …     …     …          …
94472  train_landmark_files/27610/1696867677.parquet  54756  3861       2457
94473  train_landmark_files/27610/2975578577.parquet  49608  3498       2226
94474  train_landmark_files/27610/4223702977.parquet  37440  2640       1680
94475   train_landmark_files/27610/558510995.parquet   4680   330        210
94476   train_landmark_files/27610/314634651.parquet  11232   792        504

       right_hand  frames     x_min     x_max    x_mean  y_min     y_max  \
0             357      17 -0.087367  1.199376  0.448886    0.0  2.479705
1             651      31 -0.240969  1.178582  0.419019    0.0  2.441859
2             378      18 -0.146753  1.073904  0.404705    0.0  2.518284
3             378      18 -0.069765  1.265994  0.388586    0.0  2.612595
4             294      14 -0.423106  1.303239  0.403132    0.0  2.532954
…             …     …         …         …         …    …         …
94472        2457     117 -0.117516  0.951807  0.261810    0.0  2.467156
94473        2226     106 -0.147046  0.976050  0.365789    0.0  2.550603
94474        1680      80 -0.079137  1.134678  0.466686    0.0  2.357260
94475         210      10 -0.066891  0.939062  0.431534    0.0  2.559409
```

```
94476         504      24 -0.053534  0.954912  0.390166    0.0  2.431209

         y_mean     z_min     z_max    z_mean
0      0.386801 -3.059139  3.362435 -0.055478
1      0.370088 -2.872532  1.589201 -0.058870
2      0.403468 -2.520643  1.895188 -0.038549
3      0.378294 -2.927297  2.471197 -0.022748
4      0.364148 -2.680002  2.279785 -0.042742
…           …        …         …         …
94472  0.463274 -2.706611  1.537550 -0.037171
94473  0.493334 -2.661751  1.042857 -0.050042
94474  0.496470 -2.643354  1.984523 -0.032431
94475  0.609520 -2.620925  1.503764 -0.050653
94476  0.508731 -2.105850  1.693273 -0.027899

[94477 rows x 15 columns]
```

[ ]:
```
# Assuming your DataFrame is named 'df'
metadata_df.rename(columns={'file_path': 'path'}, inplace=True)

# Display the updated DataFrame
metadata_df
```

[ ]:
```
                                                  path   face  pose  left_hand  \
0      train_landmark_files/55372/2802786652.parquet   7956   561        357
1      train_landmark_files/55372/3403106688.parquet  14508  1023        651
2      train_landmark_files/55372/1127624485.parquet   8424   594        378
3      train_landmark_files/55372/1559766834.parquet   8424   594        378
4       train_landmark_files/55372/657631983.parquet   6552   462        294
…                                                 …      …     …          …
94472  train_landmark_files/27610/1696867677.parquet  54756  3861       2457
94473  train_landmark_files/27610/2975578577.parquet  49608  3498       2226
94474  train_landmark_files/27610/4223702977.parquet  37440  2640       1680
94475   train_landmark_files/27610/558510995.parquet   4680   330        210
94476   train_landmark_files/27610/314634651.parquet  11232   792        504

       right_hand  frames     x_min     x_max    x_mean  y_min     y_max  \
0             357      17 -0.087367  1.199376  0.448886    0.0  2.479705
1             651      31 -0.240969  1.178582  0.419019    0.0  2.441859
2             378      18 -0.146753  1.073904  0.404705    0.0  2.518284
3             378      18 -0.069765  1.265994  0.388586    0.0  2.612595
4             294      14 -0.423106  1.303239  0.403132    0.0  2.532954
…             …     …        …         …         …       …        …
94472        2457     117 -0.117516  0.951807  0.261810    0.0  2.467156
94473        2226     106 -0.147046  0.976050  0.365789    0.0  2.550603
94474        1680      80 -0.079137  1.134678  0.466686    0.0  2.357260
94475         210      10 -0.066891  0.939062  0.431534    0.0  2.559409
```

```
94476         504      24 -0.053534  0.954912  0.390166    0.0  2.431209

         y_mean     z_min     z_max    z_mean
0      0.386801 -3.059139  3.362435 -0.055478
1      0.370088 -2.872532  1.589201 -0.058870
2      0.403468 -2.520643  1.895188 -0.038549
3      0.378294 -2.927297  2.471197 -0.022748
4      0.364148 -2.680002  2.279785 -0.042742
…           …        …         …         …
94472  0.463274 -2.706611  1.537550 -0.037171
94473  0.493334 -2.661751  1.042857 -0.050042
94474  0.496470 -2.643354  1.984523 -0.032431
94475  0.609520 -2.620925  1.503764 -0.050653
94476  0.508731 -2.105850  1.693273 -0.027899

[94477 rows x 15 columns]
```

[ ]:
```
# Merge the train and parquet DataFrames on the 'file_path' column
merged_df = pd.merge(metadata_df, df, on='path')
```

[ ]:
```
merged_df
```

[ ]:
```
                                                  path   face  pose  left_hand  \
0      train_landmark_files/55372/2802786652.parquet   7956   561        357
1      train_landmark_files/55372/3403106688.parquet  14508  1023        651
2      train_landmark_files/55372/1127624485.parquet   8424   594        378
3      train_landmark_files/55372/1559766834.parquet   8424   594        378
4       train_landmark_files/55372/657631983.parquet   6552   462        294
…                                                 …      …     …          …
94472  train_landmark_files/27610/1696867677.parquet  54756  3861       2457
94473  train_landmark_files/27610/2975578577.parquet  49608  3498       2226
94474  train_landmark_files/27610/4223702977.parquet  37440  2640       1680
94475   train_landmark_files/27610/558510995.parquet   4680   330        210
94476   train_landmark_files/27610/314634651.parquet  11232   792        504

       right_hand  frames     x_min     x_max    x_mean  y_min     y_max  \
0             357      17 -0.087367  1.199376  0.448886    0.0  2.479705
1             651      31 -0.240969  1.178582  0.419019    0.0  2.441859
2             378      18 -0.146753  1.073904  0.404705    0.0  2.518284
3             378      18 -0.069765  1.265994  0.388586    0.0  2.612595
4             294      14 -0.423106  1.303239  0.403132    0.0  2.532954
…             …     …        …         …         …       …         …
94472        2457     117 -0.117516  0.951807  0.261810    0.0  2.467156
94473        2226     106 -0.147046  0.976050  0.365789    0.0  2.550603
94474        1680      80 -0.079137  1.134678  0.466686    0.0  2.357260
94475         210      10 -0.066891  0.939062  0.431534    0.0  2.559409
94476         504      24 -0.053534  0.954912  0.390166    0.0  2.431209
```

```
         y_mean     z_min     z_max    z_mean  participant_id  sequence_id  \
0      0.386801 -3.059139  3.362435 -0.055478           55372   2802786652
1      0.370088 -2.872532  1.589201 -0.058870           55372   3403106688
2      0.403468 -2.520643  1.895188 -0.038549           55372   1127624485
3      0.378294 -2.927297  2.471197 -0.022748           55372   1559766834
4      0.364148 -2.680002  2.279785 -0.042742           55372    657631983
…           …         …         …         …               …           …
94472  0.463274 -2.706611  1.537550 -0.037171           27610   1696867677
94473  0.493334 -2.661751  1.042857 -0.050042           27610   2975578577
94474  0.496470 -2.643354  1.984523 -0.032431           27610   4223702977
94475  0.609520 -2.620925  1.503764 -0.050653           27610    558510995
94476  0.508731 -2.105850  1.693273 -0.027899           27610    314634651

            sign
0            any
1         vacuum
2           look
3      yesterday
4            can
…            …
94472        hot
94473       talk
94474     cowboy
94475       bird
94476        yes

[94477 rows x 18 columns]
```

[ ]:
```
# Specify the path where you want to save the CSV file
csv_file_path = '/content/drive/MyDrive/AAI521/Final Project/merged_df.csv'

# Save the DataFrame to a CSV file
merged_df.to_csv(csv_file_path, index=False)
```

[ ]:
```
# Load the DataFrame from the saved CSV file
merged_df = pd.read_csv('/content/drive/MyDrive/AAI521/Final Project/merged_df.
↪csv')
```

[ ]:
```
# Extract features and labels
feature_columns = ["face", "pose", "left_hand", "right_hand", "frames",
                   "x_min", "x_max", "x_mean", "y_min", "y_max", "y_mean",
                   "z_min", "z_max", "z_mean"]

X = merged_df[feature_columns]

label_encoder = LabelEncoder()
```

```
y = label_encoder.fit_transform(merged_df["sign"])
y = to_categorical(y)

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)
```

[ ]:
```
# Reshape the input data to include a timestep dimension
X_train_reshaped = X_train.values.reshape((X_train.shape[0], 1, X_train.
 ↪shape[1]))
X_test_reshaped = X_test.values.reshape((X_test.shape[0], 1, X_test.shape[1]))

# Build the LSTM model
model = Sequential()
model.add(LSTM(units=50, input_shape=(X_train_reshaped.shape[1],␣
 ↪X_train_reshaped.shape[2])))
model.add(Dense(units=len(label_encoder.classes_), activation='softmax'))

# Compile the model
model.compile(optimizer='adam', loss='categorical_crossentropy',␣
 ↪metrics=['accuracy'])

# Train the model
model.fit(X_train_reshaped, y_train, epochs=10, batch_size=32,␣
 ↪validation_data=(X_test_reshaped, y_test))
```

```
Epoch 1/10
2362/2362 [==============================] - 18s 5ms/step - loss: 5.5311 -
accuracy: 0.0038 - val_loss: 5.5295 - val_accuracy: 0.0038
Epoch 2/10
2362/2362 [==============================] - 10s 4ms/step - loss: 5.5292 -
accuracy: 0.0039 - val_loss: 5.5280 - val_accuracy: 0.0041
Epoch 3/10
2362/2362 [==============================] - 10s 4ms/step - loss: 5.5291 -
accuracy: 0.0039 - val_loss: 5.5299 - val_accuracy: 0.0040
Epoch 4/10
2362/2362 [==============================] - 10s 4ms/step - loss: 5.5294 -
accuracy: 0.0036 - val_loss: 5.5296 - val_accuracy: 0.0037
Epoch 5/10
2362/2362 [==============================] - 10s 4ms/step - loss: 5.5291 -
accuracy: 0.0038 - val_loss: 5.5303 - val_accuracy: 0.0039
Epoch 6/10
2362/2362 [==============================] - 10s 4ms/step - loss: 5.5293 -
accuracy: 0.0041 - val_loss: 5.5296 - val_accuracy: 0.0029
Epoch 7/10
2362/2362 [==============================] - 10s 4ms/step - loss: 5.5291 -
accuracy: 0.0039 - val_loss: 5.5313 - val_accuracy: 0.0039
```

```
Epoch 8/10
2362/2362 [==============================] - 10s 4ms/step - loss: 5.5297 -
accuracy: 0.0038 - val_loss: 5.5282 - val_accuracy: 0.0041
Epoch 9/10
2362/2362 [==============================] - 10s 4ms/step - loss: 5.5292 -
accuracy: 0.0039 - val_loss: 5.5317 - val_accuracy: 0.0040
Epoch 10/10
2362/2362 [==============================] - 10s 4ms/step - loss: 5.5291 -
accuracy: 0.0039 - val_loss: 5.5306 - val_accuracy: 0.0040
```

[ ]:
```
<keras.src.callbacks.History at 0x7b880a200fd0>
```

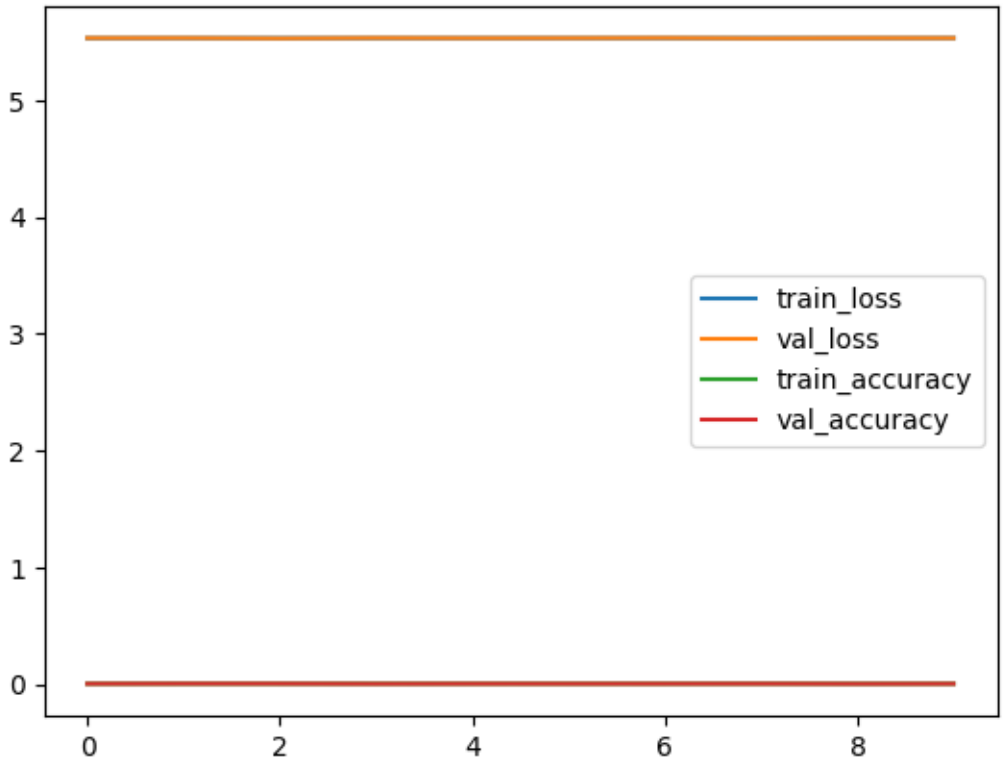
[ ]:
```
loss, accuracy = model.evaluate(X_test_reshaped, y_test)
print(f'Test Loss: {loss}, Test Accuracy: {accuracy}')
```

```
591/591 [==============================] - 2s 3ms/step - loss: 5.5306 -
accuracy: 0.0040
Test Loss: 5.530604839324951, Test Accuracy: 0.003969093784689903
```

[ ]:
```
history = model.fit(X_train_reshaped, y_train, epochs=10, batch_size=32,␣
↪validation_data=(X_test_reshaped, y_test))

# Plot training history
plt.plot(history.history['loss'], label='train_loss')
plt.plot(history.history['val_loss'], label='val_loss')
plt.plot(history.history['accuracy'], label='train_accuracy')
plt.plot(history.history['val_accuracy'], label='val_accuracy')
plt.legend()
plt.show()
```

```
Epoch 1/10
2362/2362 [==============================] - 11s 4ms/step - loss: 5.5292 -
accuracy: 0.0037 - val_loss: 5.5305 - val_accuracy: 0.0035
Epoch 2/10
2362/2362 [==============================] - 10s 4ms/step - loss: 5.5294 -
accuracy: 0.0039 - val_loss: 5.5305 - val_accuracy: 0.0043
Epoch 3/10
2362/2362 [==============================] - 10s 4ms/step - loss: 5.5293 -
accuracy: 0.0038 - val_loss: 5.5276 - val_accuracy: 0.0042
Epoch 4/10
2362/2362 [==============================] - 10s 4ms/step - loss: 5.5292 -
accuracy: 0.0043 - val_loss: 5.5310 - val_accuracy: 0.0029
Epoch 5/10
2362/2362 [==============================] - 10s 4ms/step - loss: 5.5294 -
accuracy: 0.0039 - val_loss: 5.5296 - val_accuracy: 0.0037
Epoch 6/10
2362/2362 [==============================] - 10s 4ms/step - loss: 5.5295 -
accuracy: 0.0041 - val_loss: 5.5300 - val_accuracy: 0.0035
Epoch 7/10
```

```
2362/2362 [==============================] - 10s 4ms/step - loss: 5.5292 -
accuracy: 0.0039 - val_loss: 5.5313 - val_accuracy: 0.0036
Epoch 8/10
2362/2362 [==============================] - 10s 4ms/step - loss: 5.5296 -
accuracy: 0.0041 - val_loss: 5.5280 - val_accuracy: 0.0039
Epoch 9/10
2362/2362 [==============================] - 10s 4ms/step - loss: 5.5293 -
accuracy: 0.0039 - val_loss: 5.5305 - val_accuracy: 0.0029
Epoch 10/10
2362/2362 [==============================] - 10s 4ms/step - loss: 5.5299 -
accuracy: 0.0039 - val_loss: 5.5299 - val_accuracy: 0.0035
```



[ ]: