ISOLATED SIGN LANGUAGE RECOGNITION

SPRING 2023 CS677 A2 TERM PROJECT

meher satya swaroop malina

bu id: U87601026

Boston University

**GOAL**

The goal of this project report is to present a compact isolated sign language recognition model with improved latency and the ability to run locally on mobile devices. This project is unique compared to other similar projects due to its "compact model size of < 40 MB," "reduced latency," and "ability to run on mobile devices locally." The project is a Kaggle competition aimed at helping the Popsign app identify signs made in processed videos, which will support the development of mobile apps to teach parents sign language and enable communication with their Deaf children.

**DATASET DESCRIPTION**

The dataset consists of landmarks that were extracted from raw videos with the MediaPipe holistic model. Not all of the frames necessarily had visible hands or hands that could be detected by the model.

Below is the description of each column present in the files of the dataset.

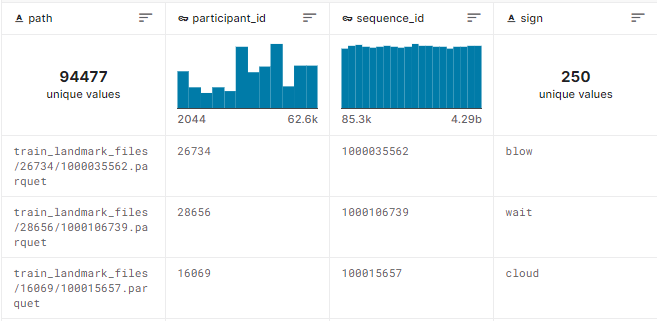
**train.csv**

path - The path to the landmark file.

participant\_id - A unique identifier for the data contributor.

sequence\_id - A unique identifier for the landmark sequence.

sign - The label for the landmark sequence.



**Landmark file**

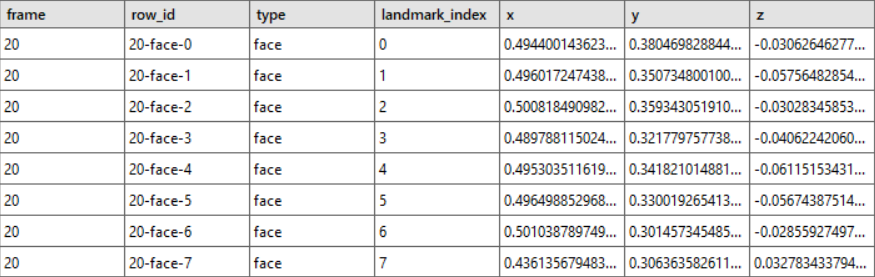
frame - The frame number in the raw video.

row\_id - A unique identifier for the row.

type - The type of landmark. One of ['face', 'left\_hand', 'pose', 'right\_hand'].

landmark\_index - The landmark index number. Details of the hand landmark locations can be found here.

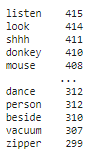
[x/y/z] - The normalized spatial coordinates of the landmark. These are the only columns that will be provided to your submitted model for inference. The MediaPipe model is not fully trained to predict depth so you may wish to ignore the z values.



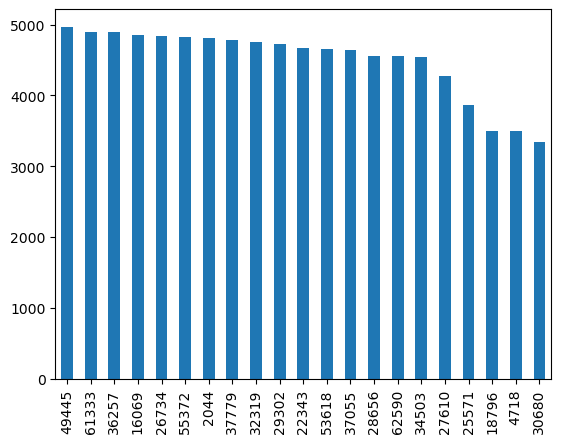
**“sign” in train.csv is the target variable.**

**EXPLORATORY DATA ANALYSIS**

* There are 21 unique participants.
* There 250 signs (classes) in the target variable.
* Each class has around 300 – 400 samples.



* Each video (landmark files), on an average, has 40 frames.
* Below is the bar plot of “participant vs no. of videos (landmark files) in the dataset”.

****

**DATA PREPROCESSING**

Target column “sign” has been label encoded.

3 separate datasets were generated using 3 various preprocessing methods.

**Preprocessing 1 (P1):**

Selected landmarks: Lips, left hand, right hand, and average of pose and face  
Selected coordinates: x, y, z

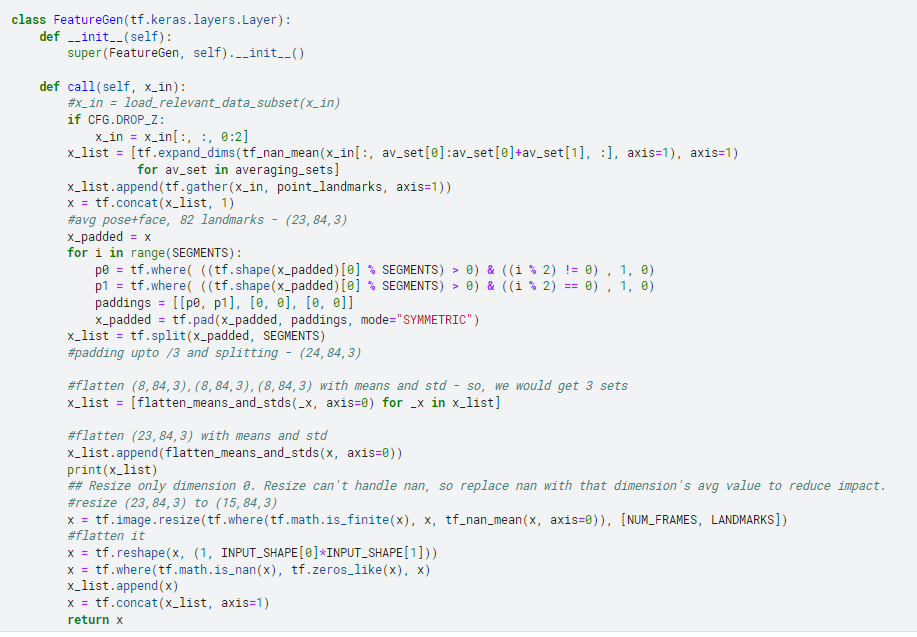
This preprocessing layer takes in a tensor of landmark data of size (n\_frames, 543, 3), where n\_frames can be any number and varies from sample to sample. The 543 represents the number of landmarks in each frame and the 3 represents the number of coordinates (x,y,z) for each landmark.

It then applies some operations to the input tensor to generate a set of features. These operations include taking the mean of subsets of the input tensor, selecting certain subsets of the input tensor, padding and splitting the input tensor, and flattening subsets of the input tensor.

The resulting feature sets are concatenated to the feature set.

The final step is to resize one of the feature sets to 15 frames, flatten it, and add it to the concatenated feature set.

**Code:**



**Preprocessing 2 (P2):**

Selected landmarks: Lips, left hand, right hand, and average of pose and face  
Selected coordinates: x, y

Same as P1 without z co-ordinate.

**Preprocessing 3 (P3):**

Selected landmarks: Lips, left hand, right hand, and pose  
Selected coordinates: x, y, z

This preprocessing layer takes in a tensor of landmark data of size (n\_frames, 543, 3), where n\_frames can be any number and varies from sample to sample. The 543 represents the number of landmarks in each frame and the 3 represents the number of coordinates (x,y,z) for each landmark.

The main goal of this layer is to prepare the input data for further processing by converting it into a fixed size tensor. The fixed size tensor is required because most deep learning models require inputs of a fixed size.

The layer first removes any empty frames in the input data. An empty frame is one in which all the landmark coordinates are missing (NaN). The layer then selects a subset of landmarks from the remaining frames that are relevant for the task at hand.

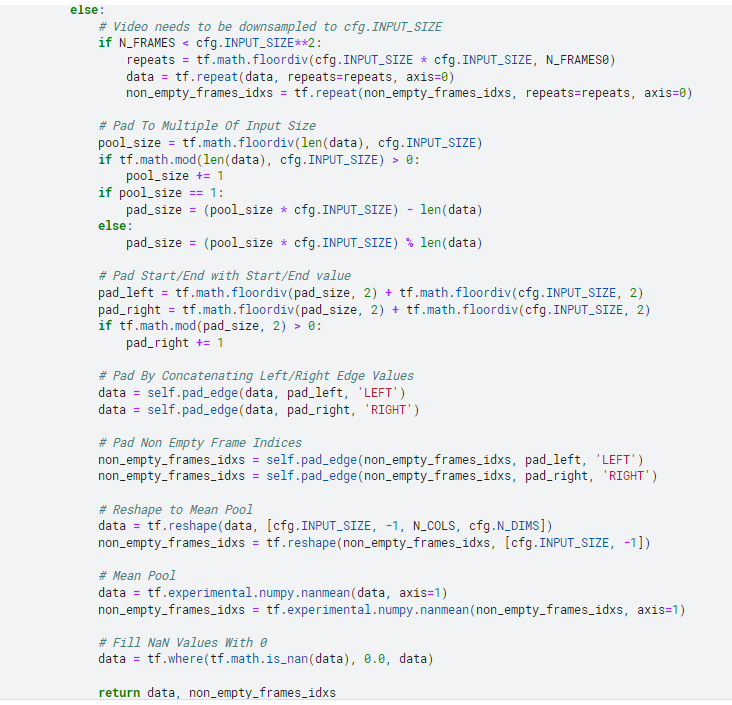
If the number of frames in the remaining data is less than 32 frames, which is a hyperparameter defined elsewhere in the code, then the data is padded with zeros to make its length equal to 32 frames. If the number of frames is greater than 32 frames, the data is down sampled using mean pooling so that the number of frames is equal to 32 frames.

If the data needs to be padded, the padding is done by concatenating the left and right edges of the data with the first and last frame, respectively. This ensures that the padding does not introduce any spurious landmarks into the data.

After padding, the data is reshaped so that it can be mean pooled across frames. The mean pooling is performed by computing the mean of each coordinate across all frames at each landmark position. Finally, any missing values (NaN) are replaced with zeros.

**Code:**





**MODEL**

**Neural Network Architecture 1 (Model 1):**

* This neural network model performs multi-class classification on a dataset with 5796 input features.
* The architecture consists of an input layer, two dense layers with 1024 and 512 nodes, respectively, and the Gelu activation function.
* Batch normalization layers are added after each dense layer to improve model performance and address the issue of internal covariate shift.
* A Flatten layer is used to convert the output from the previous layer into a one-dimensional array.
* The model is connected to an output layer consisting of 250 nodes, which uses the softmax activation function for probability prediction.
* The model is optimized using the Adam optimizer with a learning rate of 1e-3, and the loss function used is the Sparse Categorical Crossentropy.
* Three evaluation metrics are used during training: accuracy, top-5 accuracy, and top-10 accuracy.

**Code:**



**Neural Network Architecture 2 (Model 2):**

* This neural network model is designed to perform multi-class classification on a dataset with 3864 input features.
* The architecture consists of an input layer followed by three dense layers with 2048, 1024, and 512 nodes, respectively.
* Batch normalization layers are added after each dense layer to improve model performance and address the issue of internal covariate shift.
* The activation function used in all the dense layers is the Gelu activation function.
* A Flatten layer is used to convert the output from the previous layer into a one-dimensional array.
* The model is connected to an output layer consisting of 250 nodes, which uses the softmax activation function for probability prediction.
* The model is optimized using the Adam optimizer with a learning rate of 1e-3, and the loss function used is the Sparse Categorical Crossentropy.
* Three evaluation metrics are used during training: accuracy, top-5 accuracy, and top-10 accuracy.

**Code:**

****

**Transformer (Model 3):**

* The input to the model is a sequence of lip movements, left and right-hand gestures, and body poses.
* The input is passed through a custom embedding layer that converts each modality into a vector representation.
* The embedded input is then passed through a 12 headed transformer block 2 times.
* The output from the transformer blocks is pooled to create a fixed-length representation of the input sequence.
* The fixed-length representation is passed through a dense layer to classify the input into one of the specified classes.
* The model uses Sparse Categorical Crossentropy loss and AdamW optimizer with weight decay.
* The accuracy and learning rate metrics are calculated during training.

**Code:**



The models were first trained on 80% of the dataset and validated on the rest 20%. This is to get an idea on the performance of each model individually. Later, they were trained on the entire dataset and tested on the private and public test sets provided by Kaggle.

**Ensemble Learning**

Model 1 was trained on dataset generated by P1

Model 2 was trained on dataset generated by P2

Model 3 was trained on dataset generated by P3

A weighted average of the outputs of each model was then taken to combine the outputs.

* 1. \* model\_1\_outputs + 0.2 \* model\_2\_ outputs + 0.7 \* model\_3\_ outputs

**Code:**



**RESULTS**

|  |  |  |  |
| --- | --- | --- | --- |
| **Preprocessing** | **Model** | **Public Test accuracy** | **Private Test accuracy** |
| P1 | Fully connected NN  (2Dense+GELU) | 58.7% | 63.6% |
| P2 | Fully connected NN  (3Dense+GELU) | 57.3% | 63.7% |
| P3 | 12 headed Transformer | 67% | 75% |
| P1+P2+P3 | Ensemble of the above 3 | **69.7%** | **78.4%** |

**CONCLUSION**

In conclusion, I have successfully devised a concise and high-performing model for the purpose of categorizing American Sign Language (ASL) videos from a pool of 250 distinct signs, achieving a commendable test accuracy of 78%. Further enhancements to the model's performance can be attained through the incorporation of data augmentation techniques and the acquisition of additional data points.

**LINKS**

**Full code**: <https://www.kaggle.com/code/swaroopmeher/islr-deep-learning>  
**Competition:** <https://www.kaggle.com/competitions/asl-signs/overview>