Ai Sign Language Converter

Academic Supervisor

Eng.Amjad Dife

FCI

Assiut University

15 - 5 - 2024

Overview

1. Team Information
2. Task Description
3. Demo of The Running Application
4. Our Contribution
5. Data
6. Project Architecture
7. Methods
8. Results
9. End!

Overview

1. Team Information
2. Task Description
3. Demo of The Running Application
4. Our Contribution
5. Data
6. Project Architecture
7. Methods
8. Results
9. End!

Team Information

Team ID = 20

1

Youssef Ramadan Ali Ramadan (162020748)

2

Nada Abdulhamid Kamal Soudi (162020668)

3

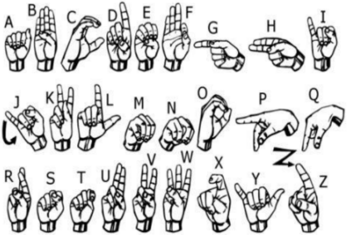
Nada Mamdouh Mohamed Ali (162020674)

Overview

1. Team Information
2. Task Description
3. Demo of The Running Application
4. Our Contribution
5. Data
6. Project Architecture
7. Methods
8. Results
9. End!

Task Description

The proposed AI sign language converter aims to easily convert sign language into text with MLP Model, which helps facilitate communication between deaf and mute people and normal people who cannot use sign language.



Overview

1. Team Information
2. Task Description
3. Demo of The Running Application
4. Our Contribution
5. Data
6. Project Architecture
7. Methods
8. Results
9. End!

Demo of The Running Application

Google Drive Link.

GitHub Link.

Overview

1. Team Information
2. Task Description
3. Demo of The Running Application
4. Our Contribution
5. Data
6. Project Architecture
7. Methods
8. Results
9. End!

Our Contribution

1. Change number of classes to be 4.
2. Replace Randomforest with MLP classifier.
3. Change images size and give the model a fixed frame and size.
4. Create dictionary of dataset with size of classes number (0: Listen, 1: Toilet, 2: Look, 3: Bored).

Overview

1. Team Information
2. Task Description
3. Demo of The Running Application
4. Our Contribution
5. Data
6. Project Architecture
7. Methods
8. Results
9. End!

Data

- The dataset consists of images of 4 classifications: (0: Listen, 1: Toilet, 2: Look, 3: Bored)

.

listen
toilet
look
bored

(a) Class 0

(b) Class 1

(c) Class 2

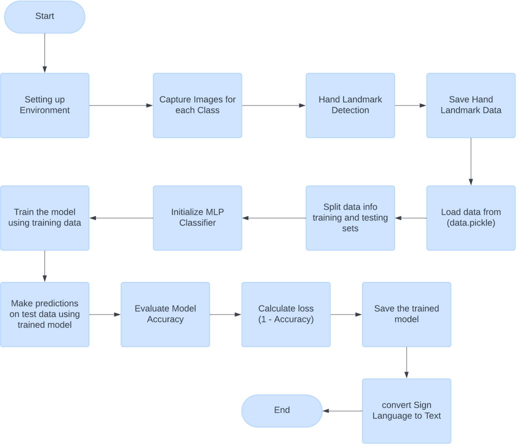
(d) Class 3

Figura: Classes of Dataset

Overview

1. Team Information
2. Task Description
3. Demo of The Running Application
4. Our Contribution
5. Data
6. Project Architecture
7. Methods
8. Results
9. End!

Project Architecture



Start
Setting up Environment
Capture Images for each Class
Hand Landmark Detection
Save Hand Landmark Data
Load data from (data.pickle)
Split data info training and testing sets
Initialize MLP Classifier
Train the model using training data
Make predictions on test data using trained model
Evaluate Model Accuracy
Calculate loss (1 - Accuracy)
Save the trained model
convert Sign Language to Text
End

Figura: MLP Architecture

Overview

1. Team Information
2. Task Description
3. Demo of The Running Application
4. Our Contribution
5. Data
6. Project Architecture
7. **Methods**
8. Results
9. End!

Methods

A. Data Collection and Image Capture:
   1. Collect image data by capturing frames from a webcam
   2. Create a directory for each class

.

B. Extract hand landmarks(data-dir):
   1. . Extract hand landmarks from images using MediaPipe Hands
   2. Prepare feature vectors for machine learning

Methods

.

C. Train and evaluate model:

   1. Load data from a pickle file
   2. Preprocess data to fixed length
   3. Split data into training and testing using train-test-split from sklearn.model-selection
   4. Initialize MLP (Multi-Layer Perceptron) classifier
   5. Train model on training data using fit
   6. Make predictions on test data
   7. Calculate accuracy, loss, and balanced accuracy
   8. Save trained MLP model

.

D. Real-time Hand Gesture Recognition:
   1. Load trained MLP model
   2. Capture frames from default camera using OpenCV's VideoCapture class
   3. S Print real-time accuracy and predicted label

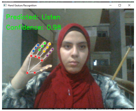
E. MLP Classifier:
   1. Input Layer: flattened image data
   2. Output Layer: corresponds to number of classes with softmax activation function
   3. Training: Backpropagation with training data adjusts model weights and biases to minimize loss function using iterative optimization.

Overview

1. Team Information
2. Task Description
3. Demo of The Running Application
4. Our Contribution
5. Data
6. Project Architecture
7. Methods
8. **Results**
9. End!

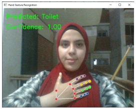
Results

1. Accuracy: 0.9 =>1
2. loss =(1 - Accuracy) => 0.1



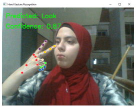
```
Predicted: Listen
Confidence: 0.98
```

(a) Result 0



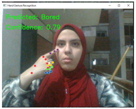
```
Predicted: Toilet
Confidence: 1.00
```

(b) Result 1



```
Predicted: Look
Confidence: 0.87
```

(c) Result 2



```
Predicted: Bored
Confidence: 0.79
```

(d) Result 3

Overview

1. Team Information
2. Task Description
3. Demo of The Running Application
4. Our Contribution
5. Data
6. Project Architecture
7. Methods
8. Results
9. End!

Thanks!