IYTE-Fall 24-25
SEDS536

Hand Gesture Recognition for Device Control

Kubra Destebasi
313011012

Problem Definition & Key Questions

Objectives: Developing a real-time gesture recognition system to detect "Thumbs Up" and "Thumbs Down" gestures to control sound settings

And create an optimal model with *a* ***smaller, yet diverse, dataset*** to reduce computational resources and training time without significantly compromising accuracy.

Key Questions:

How accurately can we detect and classify hand gestures in real-time?
What are the performance differences between YOLO for detection and KNN for classification?

Data Understanding

Description: The dataset consists of approximately 80,000 images, categorized into two classes: Thumbs Up and Thumbs Down, with 40,000 images in each class.

Data Source: Hagrid GitHub Repository (https://github.com/hukenovs/hagrid)

Dataset Characteristics Diversity:

- Hand angles,size and position.
- Distances
- Lighting Conditions
- Demographics

Selected Dataset Size: 14,000 images (7000 per class) chosen via ***random sampling***.

Data Preparation

Key Tools Used: Mediapipe for hand detection and landmark extraction.

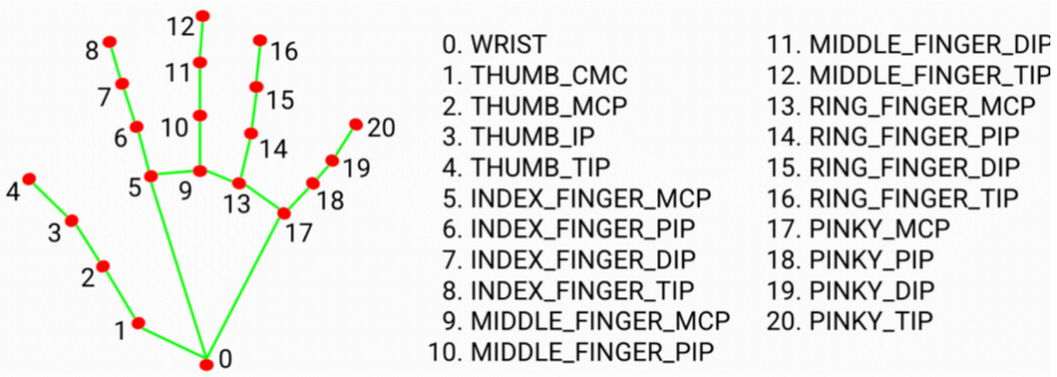
Detected Hands: Mediapipe to detect and extract hands from the images.
Reduced dataset size to ~6,100–6,300 images per class after processing.

Label Assignment:
Label 0: Thumbs Up
Label 1: Thumbs Down

Split Ratio Dataset:
Training= 0.7
Validation = 0.2
Test = 0.1

Feature Extraction for YOLO

Mediapipe Hand landmark



Generate **bounding boxes** around detected hands using Mediapipe

Bounding Box Creation:
Calculated bounding boxes based on the extremities of the hand landmarks.

Saved annotations in YOLO format (.txt files) with normalized coordinates

Change image sizes 640x640

Feature Extraction for KNN

Hand Landmark Normalization

to make the hand landmark features scale-invariant by normalizing them with respect to the wrist position.

Landmark Coordinates (X, Y, Z)

21 Landmarks: Each represented by (X, Y, Z) coordinates.

Feature Extraction for KNN

Thumb-Pinky Distance *thumb tip (L5) and pinky tip (L21)*

Calculated the Euclidean distance between the normalized coordinates of the thumb tip (index 4) and pinky tip (index 20)

Thumb-Wrist Angle *Angle (in radians) between the thumb tip (L5) and wrist (L1)*

To capture the angle formed between the thumb tip and the wrist.

providing a rotationally invariant feature.

Thumb-Pinky Vector *Vector components (X, Y, Z) between the thumb tip (L5) and pinky tip (L21).*

To capture the directional relationship between the thumb tip and pinky tip.

Overall Feature Summary

Landmark Coordinates: 63 features

Derived Features: 5 features

Class Label: 1 feature

Total Features: 69 Columns

Evaluation of KNN and YOLO

KNN achieves a higher accuracy

Metric	KNN	YOLO
Accuracy	**0.881**	0.816

performs better for gesture classification tasks in this dataset

KNN outperforms YOLO in precision

Precision	**0.939**	0.899

it makes fewer false positives. This is crucial for applications where incorrect predictions have significant consequences.

KNN also has a slight edge in recall

Recall	**0.939**	0.914

suggesting it captures more true positives than YOLO.

Evaluation of KNN and YOLO

Metric	KNN	YOLO

KNN demonstrates a superior F1-Score

F1-Score	**0.939**	0.905

reflecting a better balance between precision and recall.

KNN is significantly faster

Latency	**0.0245 seconds**	0.3954 seconds

KNN a more suitable choice for *real-time applications* on resource-constrained devices.

Hardware Requirements	Low (CPU)	High (GPU)

KNN runs on a CPU, making it accessible and efficient for deployment in environments with limited hardware capabilities.
YOLO requires a GPU for efficient inference, limiting its usability in low-resource settings.

Evaluation of KNN and YOLO

Metric	KNN	YOLO
Accuracy	**0.881**	0.816
Precision	**0.939**	0.899
Recall	**0.939**	0.914
F1-Score	**0.939**	0.905
mAP50	-	**0.960**
mAP50-95	-	0.827
Latency	**0.0245 seconds**	0.3954 seconds
Hardware Requirements	Low (CPU)	High (GPU)