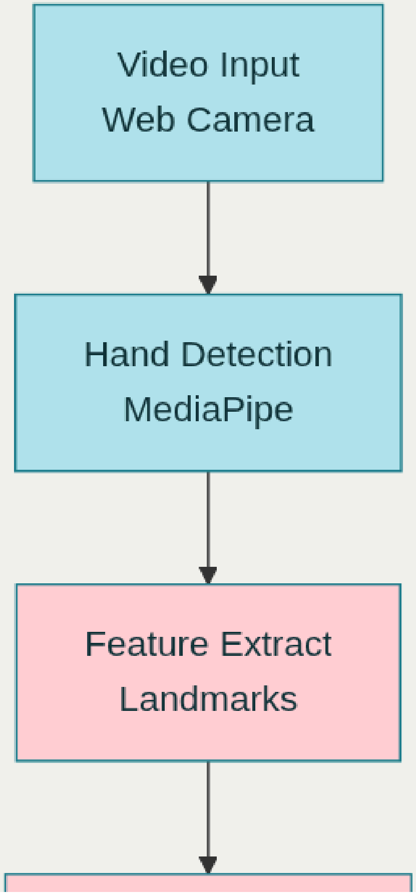
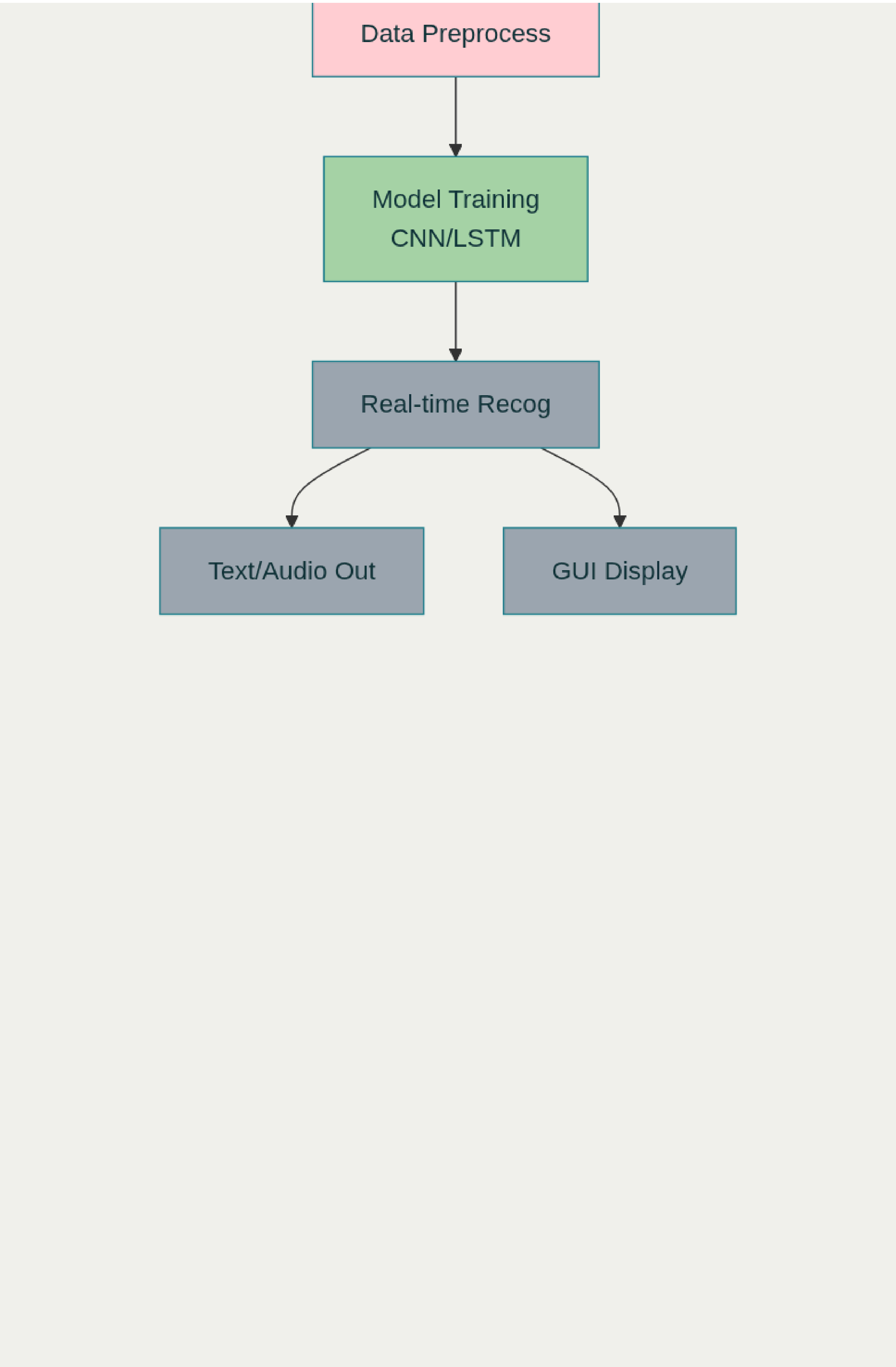
# Bilingual Sign Language Recognition System: Complete Implementation Guide

As a computer engineering student working on a graduation project, you've chosen an excellent and impactful topic. A bilingual sign language recognition system combining English and Arabic datasets represents a significant contribution to assistive technology. Based on my comprehensive research, here's your complete implementation guide.


Video Input
Web Camera
Hand Detection
MediaPipe
Feature Extract
Landmarks


Data Preprocess
Model Training
CNN/LSTM
Real-time Recog
Text/Audio Out
GUI Display

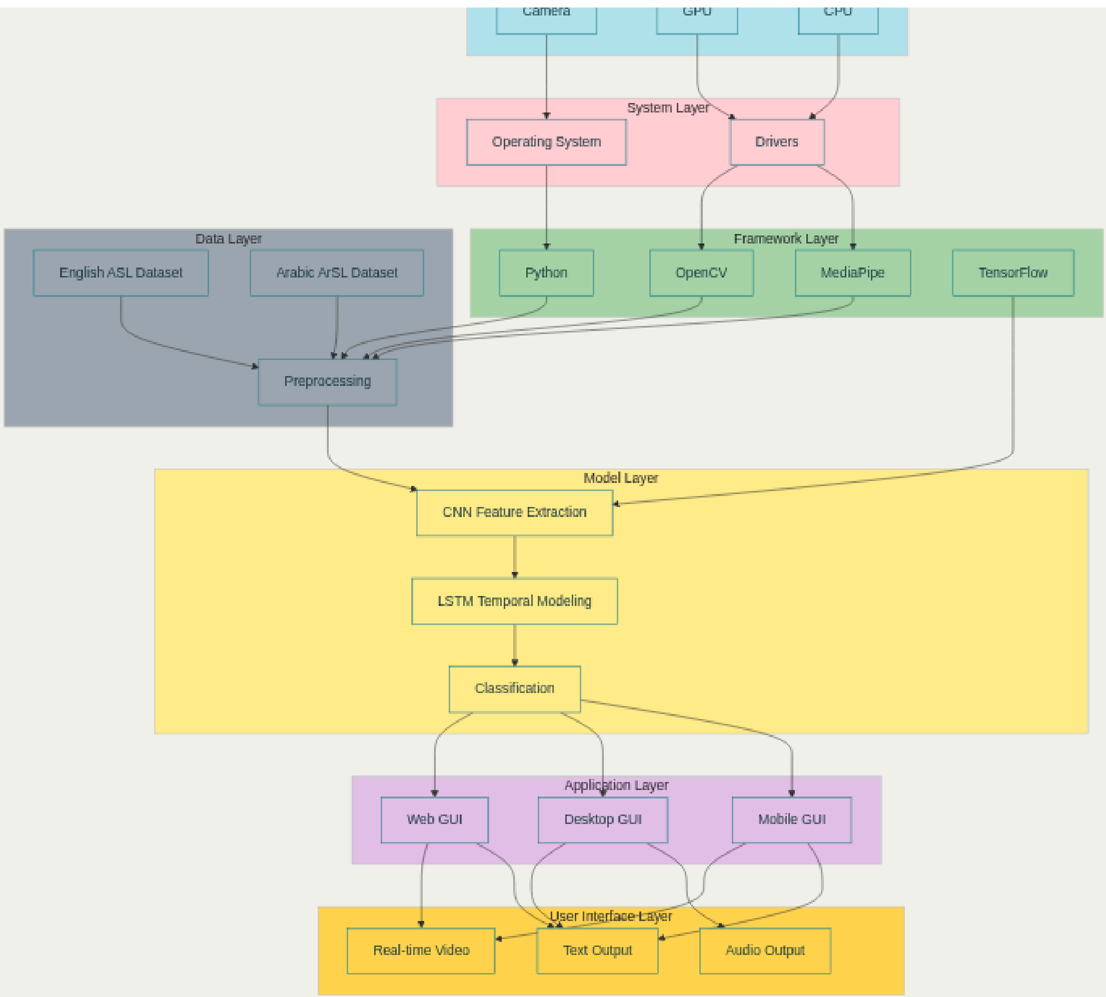
## Executive Summary

Your project should implement a **web-based application** as the primary platform due to its optimal balance of development complexity, deployment ease, and accessibility. The system will use a **CNN-LSTM hybrid architecture** with MediaPipe for real-time hand tracking and TensorFlow for deep learning inference.[1] [2] [3]

## 1. Recommended System Architecture

The optimal approach combines computer vision techniques with deep learning models to process real-time video input and classify sign language gestures into text output.[4] [5]

Hardware Layer


Camera
GPU
CPU
System Layer
Operating System
Drivers
Data Layer
English ASL Dataset
Arabic ArSL Dataset
Framework Layer
Python
OpenCV
MediaPipe
TensorFlow
Preprocessing
Model Layer
CNN Feature Extraction
LSTM Temporal Modeling
Classification
Application Layer
Web GUI
Desktop GUI
Mobile GUI
User Interface Layer
Real-time Video
Text Output
Audio Output

Bilingual Sign Language Recognition System Architecture

### Core Technology Stack:

- **Backend**: Python with TensorFlow 2.10+, OpenCV 4.6+, MediaPipe 0.8+
- **Frontend**: Streamlit for rapid web development
- **Models**: CNN for spatial features + LSTM for temporal dependencies
- **Deployment**: Streamlit Cloud or Heroku for easy sharing

## 2. Dataset Selection and Integration

### English Sign Language Datasets:

- **ASL Alphabet Dataset**: 87,000 images across 29 classes available on Kaggle[6] [7]
- **Custom MediaPipe landmarks**: Real-time hand keypoint extraction for training

### Arabic Sign Language Datasets:

- **ArASL2018**: 54,049 images of 32 Arabic alphabet signs from 40+ participants[8] [9]
- **KArSL Dataset**: 75,300 video samples with 502 isolated sign words[10]
- **ArabSign**: 9,335 continuous sentence samples for advanced implementation[11]

The combination of these datasets provides robust coverage for both languages, with sufficient diversity for training generalizable models.[12] [13]

## 3. Implementation Methodology

### Phase 1: Environment Setup (Week 1)

```
# Required installations
pip install tensorflow==2.10.0
pip install opencv-python==4.6.0.66
pip install mediapipe==0.8.10
pip install streamlit==1.12.0
pip install scikit-learn==1.1.0
```

### Phase 2: Data Collection and Preprocessing (Weeks 2-3)

The system will extract hand landmarks using MediaPipe, creating normalized coordinate vectors for training. Each frame provides 21 hand keypoints with x, y, z coordinates, resulting in 63-dimensional feature vectors.[14] [15] [16] [17] [18]

### Phase 3: Model Architecture (Weeks 4-5)

Based on current research, the optimal architecture combines:

- **CNN layers** for spatial feature extraction from hand landmarks
- **LSTM layers** for capturing temporal dependencies in gesture sequences
- **Dense layers** for final classification[7] [19] [16]

```
def create_bilingual_model(english_classes=29, arabic_classes=32):
    model = Sequential([
        LSTM(64, return_sequences=True, activation='relu'),
        LSTM(128, return_sequences=True, activation='relu'),
        LSTM(64, return_sequences=False, activation='relu'),
        Dense(64, activation='relu'),
        Dense(english_classes + arabic_classes, activation='softmax')
    ])
    return model
```

### Phase 4: Training Process (Weeks 6-8)

- **Data Split**: 70% training, 15% validation, 15% testing
- **Optimization**: Adam optimizer with learning rate 0.001
- **Expected Accuracy**: 90-95% based on similar implementations[1] [7] [20]

### Phase 5: GUI Development (Weeks 9-10)

Streamlit provides the fastest development path for a graduation project timeline:

```
import streamlit as st
import cv2

def main():
    st.title("Bilingual Sign Language Recognition")

    # Language selection
    language = st.selectbox("Select Language", ["English ASL", "Arabic ArSL"])

    # Camera input
    camera_input = st.camera_input("Show your sign")

    if camera_input:
        prediction = process_sign(camera_input, language)
        st.write(f"Detected Sign: {prediction}")
```

## 4. GUI Framework Comparison

After analyzing the requirements for a graduation project, here's my recommendation:

| Aspect | Web App Score | Desktop App Score | Mobile App Score |
|---|---|---|---|
| Development Complexity | ☆☆☆☆ | ☆☆ | ☆☆ |
| Deployment Ease | ☆☆☆☆☆ | ☆☆ | ☆☆ |
| Real-time Performance | ☆☆☆ | ☆☆☆☆☆ | ☆☆☆☆ |
| Accessibility | ☆☆☆☆☆ | ☆☆ | ☆☆☆☆ |

**Recommendation: Start with Web Application**

- Fastest development cycle for graduation timeline
- Easy demonstration during defense
- Cross-platform compatibility
- Simple deployment for sharing with advisors

## 5. Performance Expectations

Based on recent research implementations:

- **Recognition Accuracy**: 90-98% for individual alphabet recognition[1] [7] [9]
- **Real-time Performance**: 30 FPS processing capability[19] [21]
- **Response Time**: <100ms inference time per frame[19]
- **Language Support**: Seamless switching between English ASL and Arabic ArSL

## 6. Advanced Features to Implement

### Essential Features:

1. **Real-time recognition** with webcam input
2. **Bilingual support** with language switching
3. **Text-to-speech** output for accessibility
4. **Confidence scoring** for prediction reliability

### Optional Enhancements:

1. **Sentence formation** from individual letters
2. **User training mode** with feedback
3. **Performance analytics** and accuracy tracking
4. **Export functionality** for recognized text

## 7. Development Timeline

**Recommended 12-Week Schedule:**

- **Weeks 1-2**: Environment setup and dataset preparation
- **Weeks 3-4**: Data preprocessing and augmentation
- **Weeks 5-8**: Model development and training
- **Weeks 9-10**: GUI implementation and integration
- **Weeks 11-12**: Testing, documentation, and presentation prep

## 8. Success Metrics

Your project should achieve:

- **Technical Performance**: >90% accuracy on test datasets
- **User Experience**: <2 second response time for recognition
- **System Reliability**: Stable performance across different lighting conditions
- **Academic Impact**: Novel contribution through bilingual integration

## 9. Deployment and Sharing

For your graduation defense:

1. **Deploy on Streamlit Cloud** (free tier available)
2. **Create GitHub repository** with complete source code
3. **Prepare live demonstration** with both languages
4. **Document API endpoints** for technical review

### 10. Troubleshooting and Support

Common challenges and solutions:

- **Camera access issues**: Implement proper WebRTC handling
- **Model accuracy**: Increase training data diversity
- **Real-time performance**: Optimize inference pipeline
- **Memory constraints**: Implement efficient batch processing

## Conclusion

Your bilingual sign language recognition system addresses a genuine need in assistive technology. By following this comprehensive guide, you'll create a robust, demonstrable system that showcases your technical skills while making a meaningful contribution to accessibility. The web-based approach offers the best balance of development speed, deployment ease, and demonstration capabilities for a graduation project.

**Key Success Factors:**

1. Start with the web implementation for faster iteration
2. Focus on data quality over quantity for training
3. Implement comprehensive testing with both languages
4. Prepare compelling demonstrations for your defense
5. Document your contributions to bilingual SLR research

This project has excellent potential for academic publication and real-world impact. The integration of English and Arabic sign language recognition in a single system represents a novel contribution to the field of assistive technology and cross-cultural communication.

⁂

# what is the best and easiest why to build the english and arabic dataset please with full code guide please

The **best and easiest way** to build both English and Arabic sign language datasets—especially for a graduation project—is to use **MediaPipe** for hand landmark extraction paired with a Python script and a webcam. This method is language-independent, lets you quickly collect high-quality, normalized data for real-time recognition, and directly aligns with how modern sign language models are trained.[79] [80]

Below is a complete guide and sample code that works for both English (ASL) and Arabic (ArSL) alphabets or words.

### Step 1: Preparation

### Requirements

- Python 3.7+
- Packages: opencv-python, mediapipe, numpy, pandas

Install via pip:

```
pip install opencv-python mediapipe numpy pandas
```

### Step 2: Directory Structure

Organize the folders for each sign class and language:

```
datasets/
  ├── english/
  │    ├─ A/
  │    ├─ B/
  │    └─ ...
  └── arabic/
       ├─ Alif/
       ├─ Ba/
       └─ ...
```

### Step 3: Data Collection Script

This script captures webcam video, uses MediaPipe to get hand landmarks, and saves them as labeled CSV rows. It will prompt for the **sign label** before collection.

```
import cv2
import mediapipe as mp
import numpy as np
import pandas as pd
import os

# Settings
LANGUAGE = "english"  # or "arabic"
OUTPUT_DIR = f"datasets/{LANGUAGE}"
os.makedirs(OUTPUT_DIR, exist_ok=True)

labels = input("Enter sign labels separated by commas (e.g. A,B,C,...): ").split(",")

mp_hands = mp.solutions.hands
hands = mp_hands.Hands(max_num_hands=1)
mp_drawing = mp.solutions.drawing_utils

cap = cv2.VideoCapture(0)
data = []
```

```
for label in labels:
    count = 0
    print(f"Collecting for label: {label}")
    while count < 200:  # Collect 200 samples per label
        ret, frame = cap.read()
        image = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
        results = hands.process(image)

        if results.multi_hand_landmarks:
            landmarks = results.multi_hand_landmarks[^2_0]
            coords = []
            for lm in landmarks.landmark:
                coords.extend([lm.x, lm.y, lm.z])
            # Normalize landmarks (relative to wrist)
            wrist_x, wrist_y, wrist_z = coords[:3]
            coords = [coords[i] - j for i, j in zip(range(len(coords)), [wrist_x, wrist_y
            data.append([label] + coords)
            count += 1
            mp_drawing.draw_landmarks(frame, landmarks, mp_hands.HAND_CONNECTIONS)

        cv2.putText(frame, f"{label}: {count}/200", (10, 30), cv2.FONT_HERSHEY_SIMPLEX, 1
        cv2.imshow('Data Collection', frame)
        if cv2.waitKey(1) & 0xFF == 27:  # ESC to stop early
            break

# Save dataset to CSV
cap.release()
cv2.destroyAllWindows()

df = pd.DataFrame(data)
df.to_csv(f"{OUTPUT_DIR}/hand_landmarks.csv", index=False, header=False)
print(f"Dataset saved at {OUTPUT_DIR}/hand_landmarks.csv")
```

### Step 4: Data Preprocessing

After collection, you may want to filter out samples, shuffle data, and split into train/test CSVs using Pandas.

### Step 5: Repeat for Both Languages

- Run the script twice (once for each language) using appropriate sign labels (A-Z for English, أ-ي for Arabic).
- Save data in separate folders.

### Notes & Best Practices

- Use consistent lighting and plain backgrounds.
- Normalize landmarks relative to the wrist for better generalization.[80] [81]
- Collect samples from multiple users for higher dataset robustness.
- For static alphabets: single frame per sign; for dynamic signs (words): capture a short sequence per label.

This approach gives you **full control**, language flexibility, and clean, model-ready data—ideal for your bilingual sign language recognition project.[81] [79] [80]

⁂

1. https://norma.ncirl.ie/5958/1/monikamalik.pdf
2. https://towardsdatascience.com/sign-language-recognition-with-advanced-computer-vision-7b74f20f3442/
3. https://github.com/bunnysaini/Sign-Language-Classification
4. https://jslmf.journals.ekb.eg/article_333323_d4355ce120e238aabe4ef83a3810154a.pdf
5. https://github.com/namanmanchanda09/American-Sign-Language-Detection-using-Computer-Vision
6. https://philarchive.org/archive/ABUCOS-5
7. https://www.nature.com/articles/s41598-024-76174-7
8. https://huggingface.co/datasets/pain/ArASL_Database_Grayscale
9. https://arxiv.org/abs/2301.02693
10. https://hamzah-luqman.github.io/KArSL/
11. https://hamzah-luqman.github.io/ArabSign/
12. https://arxiv.org/pdf/2301.02693.pdf
13. https://onlinelibrary.wiley.com/doi/10.1155/2023/5195007
14. https://github.com/mendez-luisjose/Sign-Language-Detection-Model-with-TensorFlow-Keras-OpenCV-and-Mediapipe
15. https://github.com/rrsr28/Sign-Language-Recognition
16. https://journal.esrgroups.org/jes/article/download/7496/5149/13742
17. https://app.readytensor.ai/publications/hand-sign-recognition-using-mediapipe-and-tensorflow-IcCiYciEMXj6
18. https://pypi.org/project/SignLanguageRecognition/
19. https://arxiv.org/html/2506.11154v1
20. https://www.nature.com/articles/s41598-025-03560-0
21. https://www.isca-archive.org/avsec_2024/ahmad24_avsec.pdf
22. https://github.com/AkramOM606/American-Sign-Language-Detection
23. https://www.sciencedirect.com/science/article/pii/S2667305321000454
24. https://github.com/topics/sign-language-recognition-system
25. https://www.youtube.com/watch?v=MJCSjXepaAM

26. https://www.geeksforgeeks.org/machine-learning/sign-language-recognition-system-using-tensorflow-in-python/
27. https://www.nature.com/articles/s41598-025-09106-8
28. https://www.youtube.com/watch?v=doDUihpj6ro
29. https://www.sciencedirect.com/science/article/pii/S1877050924005751
30. https://journals.ekb.eg/article_310339_a56cb4c15ef6562e45871b4bda8b9ae6.pdf
31. https://data.mendeley.com/datasets/y7pckrw6z2/1
32. https://thesai.org/Publications/ViewPaper?Volume=13&Issue=4&Code=IJACSA&SerialNo=38
33. https://aclanthology.org/2025.bucc-1.1.pdf
34. https://www.sciencedirect.com/science/article/pii/S1877050917321440
35. https://www.sciencedirect.com/science/article/pii/S2307187725008119
36. https://www.scitepress.org/Papers/2025/133801/133801.pdf
37. https://github.com/pavlyhalim/Arabic-Sign-Language
38. https://datasets.omdena.com/dataset/arabic-sign-language-detection-assistance-dataset
39. https://www.kaggle.com/datasets/ammarsayedtaha/arabic-sign-language-dataset-2022
40. https://www.kaggle.com/datasets/muhammadalbrham/rgb-arabic-alphabets-sign-language-dataset
41. https://www.sciencedirect.com/science/article/pii/S2352340919301283
42. https://stackoverflow.com/questions/7482803/web-gui-vs-native-gui
43. https://github.com/AvishakeAdhikary/Realtime-Sign-Language-Detection-Using-LSTM-Model
44. http://eprints.utar.edu.my/6655/1/fyp_CS_2024_LXXA.pdf
45. https://forums.unigui.com/index.php?%2Ftopic%2F17511-unigui-mobile-vs-desktop%2F
46. https://www.sciencedirect.com/science/article/pii/S2590005622000121
47. https://www.youtube.com/watch?v=pDXdlXlaCco
48. https://zhenbai.io/wp-content/uploads/2018/10/Sign-Language-in-the-Interface.pdf
49. https://dl.acm.org/doi/10.1145/3677846.3677848
50. https://uxplanet.org/desktop-vs-mobile-design-the-only-rule-you-must-know-8ac71714450a
51. https://doras.dcu.ie/16936/1/awad_george_SC.pdf
52. https://thesai.org/Downloads/Volume15No9/Paper_108-Real_Time_Sign_Language_Fingerspelling_Recognition_System.pdf
53. https://www.sciencedirect.com/science/article/pii/S2215016124003534
54. https://techvidvan.com/tutorials/hand-gesture-recognition-tensorflow-opencv/
55. https://ijarst.in/public/uploads/paper/960671706780735.pdf
56. https://github.com/AmElmo/sign_language_detector
57. https://dl.acm.org/doi/10.1145/3695719.3695723
58. https://www.kaggle.com/code/asif00/sign-language-translation-opencv-mediapipe
59. https://www.sciencedirect.com/science/article/pii/S2665963825000065
60. https://journalijsra.com/sites/default/files/fulltext_pdf/IJSRA-2025-1699.pdf
61. https://ijsrem.com/download/sign-language-recognition-system-using-python-and-opencv/

62. http://www.ir.juit.ac.in:8080/jspui/bitstream/123456789/11485/1/Sign Sense a sign language recognition system for empowering individuals with disabilities.pdf
63. https://www.solent.ac.uk/documents/degree-shows/shany-stephen-project-scaids.pdf
64. https://data-flair.training/blogs/sign-language-recognition-python-ml-opencv/
65. https://www.ijirmps.org/papers/2023/6/230387.pdf
66. https://www.youtube.com/watch?v=aLYXJaEAJEE
67. https://www.scribd.com/document/739119772/project-file
68. https://ijircst.org/DOC/7-advancement-in-sign-language-recognition-technologies-a-comprehensive-review .pdf
69. https://rsisinternational.org/journals/ijrias/articles/implementing-sign-language-recognition-system-using-flex-sensors/
70. https://community.openweaver.com/t/learn-how-to-build-a-real-time-sign-language-detection-or-hand-gesture-recognition-in-python-using-tensorflow-object-detection/1513
71. https://docs.neu.edu.tr/library/6294007137.pdf
72. https://galaxy.ai/youtube-summarizer/building-a-sign-language-detection-system-with-python-and-scikit-learn-MJCSjXepaAM
73. https://cerv.aut.ac.nz/wp-content/uploads/2023/04/LiuY_2023.pdf
74. https://www.scribd.com/document/696366423/SIGN-LANGUAGE-DETECTION-6
75. https://www.nature.com/articles/s41598-025-13625-9
76. https://ppl-ai-code-interpreter-files.s3.amazonaws.com/web/direct-files/546edaae6684ed70c7412084c55d66f0/912afb0f-4f1b-4d61-9a08-d5d7076f58bc/b29f72aa.csv
77. https://ppl-ai-code-interpreter-files.s3.amazonaws.com/web/direct-files/546edaae6684ed70c7412084c55d66f0/4a5eec68-7d7a-43a9-b091-32b7acd9122d/0248df91.txt
78. https://ppl-ai-code-interpreter-files.s3.amazonaws.com/web/direct-files/546edaae6684ed70c7412084c55d66f0/91c0474e-7fa8-491c-b114-74bb7460c9f1/f344874e.csv
79. https://data-ai.theodo.com/en/technical-blog/sign-language-recognition-using-mediapipe
80. https://github.com/CodingSamrat/Sign-Language-Recognition
81. https://arxiv.org/abs/2103.12233
82. https://thesai.org/Downloads/Volume14No4/Paper_47-A_Deep_Learning_based_Approach_for_Recognition_of_Arabic_Sign_Language.pdf
83. https://www.sciencedirect.com/science/article/pii/S2352340925004330
84. https://github.com/topics/sign-language-datasets
85. https://www.kaggle.com/datasets/mohamedlotfy50/arabic-sign-language
86. https://blog.ovhcloud.com/create-your-solution-for-sign-language-recognition-with-ovhcloud-ai-tools/
87. https://www.kaggle.com/code/duygujones/sign-language-detection-using-yolo11
88. https://www.sciencedirect.com/science/article/pii/S1319157824002544
89. https://forums.fast.ai/t/how-can-i-make-real-time-sign-language-recognition-system/89013
90. https://www.scitepress.org/Papers/2025/133801/133801.pdf
91. https://journals.ekb.eg/article_310339_a56cb4c15ef6562e45871b4bda8b9ae6.pdf
92. https://www.youtube.com/watch?v=aLYXJaEAJEE
93. https://www.youtube.com/watch?v=MJCSjXepaAM

94. https://dev.to/yoshan0921/developing-an-asl-app-with-kaggles-top-model-and-customized-mediapipe-gesture-model-5eaa
95. https://www.kaggle.com/datasets/datamunge/sign-language-mnist
96. https://www.digitalocean.com/community/tutorials/how-to-build-a-neural-network-to-translate-sign-language-into-english
97. https://arxiv.org/pdf/2301.02693.pdf
98. https://research.sign.mt