ASL Recognition System: Bridging Gaps in Communication Accessibility

A Thesis Submitted in partial fulfillment of the Requirements of the Renée Crown University Honors Program at Syracuse University

Carlo F. Pisacane

Candidate for Bachelor's of Science in Computer Science
and Renée Crown University Honors
May 2025

Honors Thesis in Computer Science

Thesis Advisor: ____________________
Dr. Nadeem Ghani, Assistant Teaching Professor

Thesis Reader: ____________________
Dr. João Paulo Marum, Assistant Teaching Professor

Honors Director:____________________
Dr. Danielle Smith, Director

Abstract

A concise summary of your research, covering the problem (gaps in existing ASL recognition systems), your proposed solution (ASL recognition tool), methodology, key findings, and conclusions.

Thesis Advisor: Nadeem Ghani
Title: Assistant Teaching Professor, Engineering and Computer Science

Acknowledgments

I am deeply grateful for the support and guidance I have received throughout my academic journey at Syracuse University. I would like to extend my sincere thanks to my advisor, Prof. Nadeem Ghani, whose mentorship was instrumental in shaping my research and academic pursuits. My gratitude also goes to my thesis reader, Prof. João Marum, for his valuable insights and constructive feedback.

I am particularly thankful to Robin Smith, my Honors advisor, for motivating and encouraging me over the past four years during my time in the Honors program and throughout the thesis development process. Their support helped me overcome challenges and remain focused on my goals.

Finally, I am forever indebted to my friends and family, whose encouragement and belief in me have been a constant source of inspiration.

Preface

Why I chose this topic, personal insights, and what I learned, reference: CS example thesis.pdf (Kyle Maiorana)

Contents

List of Figures

List of Excerpts

Chapter 1

Introduction

Communication is a fundamental human need, and for Deaf and hard-of-hearing individuals, American Sign Language (ASL) serves as a primary mode of expression. ASL is a rich and complex visual language based on hand shape, movement, and nonmanual markers such as facial expressions. Studies emphasizing Deaf-centric design have shown that effective ASL tools must respect the cultural and linguistic practices of the Deaf community (Hibbard et al., 2020 [1]).

Technology has evolved from the past to make communication more accessible, including captioning services, text-based messaging, and video relay services. However, these solutions often rely on real-time interpreters or a shared written language, which can be limiting in spontaneous, in-person conversations.

In recent years, advances in computer vision and machine learning have opened new avenues for automated sign language recognition. Leveraging the power of advanced algorithms and increasingly pervasive hardware (e.g., mobile phone cameras, webcams), engineers and researchers hope to create machines that can recognize ASL hand gestures in real time. While some progress has been made in recognizing static signs or alphabets (Debnath and Joe, 2024 [3]), challenges remain, especially regarding vocabulary expansion, nonmanual features, and real-world performance (Falvo et al., 2020 [2]). The present work focuses on addressing these challenges by developing a user-centered ASL recognition tool that emphasizes accuracy, speed, and accessibility.

1.1 Research Problem

Current ASL recognition systems have several limitations that make them impractical. Some systems recognize only a limited set of signs, which restricts real-world applicability. Others do not account for nonmanual signs, such as facial expressions or head tilts, which are essential in ASL for the expression of tone, grammatical markers, and emotional context. Additionally, some tools demand specialized sensors that are expensive or inconvenient, thus deterring widespread use.

There is also a broad gap in designing user interfaces that are responsive to the needs and desires of Deaf individuals. Inaccurate calibration procedures, variability of performance under changing lighting conditions, or excessive latency can lower the usability of a system. These barriers compound to inhibit the real-world deployment potential of ASL recognition technology in everyday communication (Falvo et al., 2020 [2]).

1.2 Research Objectives

The overall goal of this thesis is to design a real-time ASL recognition system using computer vision and machine learning techniques. Specifically, the system must offer high recognition accuracy. The second aim is to enhance user experience by developing an accessible interface that is easy to install and requires minimal calibration or dedicated hardware. Additionally, the system needs better performance in the range of lighting and backgrounds found in real world environments. Finally, the framework needs to be extensible for future additions of other signs, dynamic gestures, and nonmanual signals. By focusing on these core objectives, the system aims to be a foundation for broader applications in education, assistive technology, and inclusive communication devices.

1.3 Significance of the Study

This project holds the potential for shattering communication barriers among the Deaf and hard-of-hearing and for providing meaningful resources for hearing individuals who wish to learn ASL. An effective real-time ASL recognition system can facilitate communication more effectively in public places, schools, and workplaces, particularly where access to interpreters may not be readily available. It can also serve as an interactive learning tool for ASL learners, offering instant feedback on handshapes to facilitate language learning. The proposed framework can further be extended to encompass the full richness of sign languages, thereby enabling future research studies. By prioritizing usability and involving Deaf/ASL communities in the design, this research underscores the importance of user-centered solutions.

1.4 Thesis Structure This thesis is organized into five main chapters:

1. **Chapter 1: Introduction**

   Provides the background and context of ASL recognition, defines the research problem, outlines objectives, and explains the study's significance.

2. **Chapter 2: Literature Review**

   Examines existing ASL recognition systems, machine learning techniques, and user-centered design principles. Identifies gaps in the current research and sets the stage for the proposed approach.

3. **Chapter 3: Software and Application**

   Details the methodology, including data collection, model architecture, and real-time inference pipeline. Discusses the design choices and rationale behind the system's implementation.

4. **Chapter 4: Results**

   Presents empirical findings, including model performance metrics, real-time testing results, and user feedback. Analyzes both quantitative and qualitative data.

5. **Chapter 5: Conclusion**

   Summarizes key insights, highlights contributions, and suggests avenues for future work. Reflects on the system's potential impact on accessibility and communication technologies.

Chapter 2

Literature Review

2.1 Introduction

This literature review surveys various methods and technologies employed in American Sign Language (ASL) recognition systems. The goal is to establish how our approach, aimed at real-time recognition and user-friendly interaction fits into the existing body of work. By examining both hardware-based and vision-based solutions, we highlight the fundamental challenges and opportunities in creating accessible tools for Deaf and hard-of-hearing communities.

2.2 Overview of Existing ASL Recognition Systems

Recent developments in sign language recognition often revolve around three main technological streams:

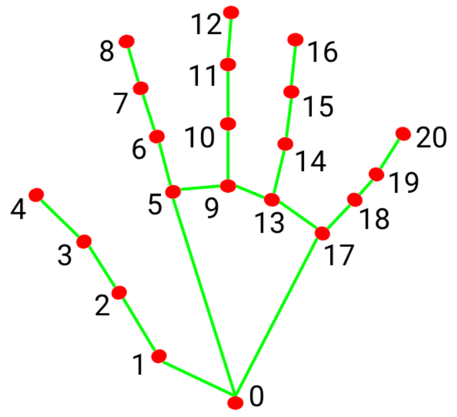
- Computer Vision Approaches
- Depth Sensor Approaches
- Wearable Technology Approaches

Each of these streams has strengths and limitations related to cost, accuracy, ease of deployment, and user comfort. This section reviews notable research in these areas and provides the groundwork for our own system's design choices.

2.2.1 Computer Vision Approaches

A substantial portion of ASL recognition research leverages standard RGB cameras and a variety of computer vision techniques. Traditional pipelines often involve skin detection, feature extraction, and hand segmentation before proceeding to classification. More recently, robust libraries like *OpenCV* and frameworks such as *MediaPipe* have significantly simplified and improved hand detection and tracking.

One notable example is the *MediaPipe Hands* solution, an open-source tool by Google[1]. MediaPipe Hands tracks 21 key landmarks on each hand, as illustrated in Figure 1. This framework provides real-time tracking even under challenging conditions such as varied lighting or partial occlusions, which is crucial for the fluid and rapid gestures of sign language.



0. WRIST
1. THUMB_CMC
2. THUMB_MCP
3. THUMB_IP
4. THUMB_TIP
5. INDEX_FINGER_MCP
6. INDEX_FINGER_PIP
7. INDEX_FINGER_DIP
8. INDEX_FINGER_TIP
9. MIDDLE_FINGER_MCP
10. MIDDLE_FINGER_PIP
11. MIDDLE_FINGER_DIP
12. MIDDLE_FINGER_TIP
13. RING_FINGER_MCP
14. RING_FINGER_PIP
15. RING_FINGER_DIP
16. RING_FINGER_TIP
17. PINKY_MCP
18. PINKY_PIP
19. PINKY_DIP
20. PINKY_TIP

Figure 1: MediaPipe Hands landmarks. Each point corresponds to a specific joint or fingertip.

Researchers have successfully integrated MediaPipe's tracking features into sign language systems to reduce the complexity of manual feature engineering. For instance, Debnath and Joe [3] demonstrate a real-time ASL detection setup where MediaPipe is employed to extract hand pose information, thereby reliably differentiating between common ASL hand shapes and movements.

2.2.2 Depth Sensor Approaches

Beyond standard RGB cameras, *depth sensors* capture three-dimensional information about hand positions. Notably, Avola et al. [4] illustrate how the Leap Motion Controller can track the skeletal structure of the hand with fine-grained accuracy, enabling precise

[1]`https://github.com/google-ai-edge/mediapipe/blob/master/docs/solutions/hands.md`

measurement of joint angles. This capability is beneficial for distinguishing similar signs that differ only slightly in finger placement or orientation.

2.2.3 Wearable Technology Approaches

Finally, some research efforts rely on *wearable devices*, such as glove-based sensors, to record hand movements. These systems, often equipped with inertial measurement units (IMUs) or bend sensors, can capture motion data in three dimensions. For example, Stefanidis et al. [5] discuss the use of 3D technologies in sign language applications through wearable platforms. While potentially more accurate, such devices can be costly or cumbersome, limiting their broad adoption for everyday use.

2.3 Summary

In summary, contemporary ASL recognition solutions range from purely vision-based approaches (e.g., MediaPipe Hands combined with OpenCV for live detection) to more specialized hardware solutions (e.g., depth sensors and wearable gloves). Among these, live vision-based detection using MediaPipe and OpenCV offers the optimal balance of high accuracy, cost-effectiveness, and ease of integration, making it the best choice for designing robust, real-time ASL recognition systems that can be easily adopted in real-world contexts.

2.4 Machine Learning Approaches for Gesture Recognition

Advances in deep learning have been essential in enhancing sign language recognition by addressing both spatial and temporal complexities. In our work and as demonstrated in previous studies (Papastratis et al., 2021 [9] and Ru and Sebastian, 2023 [10]), a combination of Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and Long Short-Term Memory (LSTM) networks has been used to capture static hand features as well as the dynamic information inherent in continuous gestures.

2.4.1 Convolutional Neural Networks (CNNs) for Static Sign Recognition

CNNs excel at extracting spatial features from images or individual video frames. Their layered architecture enables the learning of hierarchical representations that can effectively distinguish subtle differences in hand shape and orientation. In the context of ASL recognition, CNNs have been successfully used to classify isolated, static signs before further processing is applied (Debnath and Joe, 2024 [3]).

2.4.2 Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTMs) for Dynamic Sign Recognition

Dynamic signs which involve motion and transitions require models that can capture temporal dependencies. Traditional RNNs are naturally suited for sequential data, but they often encounter difficulties when processing long input sequences (Avola et al., 2019 [4]; Liu et al., 2016 [6]). LSTM networks, introduced to overcome these limitations, incorporate memory cells and gating mechanisms that enable them to retain and update information over extended periods (Hochreiter and Schmidhuber, 1997 [7]; Colah's Blog, 2015 [8]). These capabilities make LSTMs particularly effective for recognizing continuous ASL gestures.

Hybrid architectures that combine CNNs and LSTMs have also proven effective. In such systems, CNNs first extract deep features from each frame, and these features are then processed by LSTMs to capture the sequential dynamics of the gesture (Papastratis et al., 2021 [9]; Ru and Sebastian, 2023 [10]). This two-stage approach enhances accuracy, particularly in real-world scenarios where both spatial and temporal information are crucial.

2.4.3 Handling Static vs. Dynamic Signs

The choice of model architecture is dictated by the nature of the sign:

- **Static Signs:** When the sign consists of a fixed hand posture, CNNs can be employed alone to capture the necessary spatial features.

- **Dynamic Signs:** For gestures that involve motion, models such as RNNs and, more effectively, LSTMs are essential. Their ability to model long-term dependencies allows them to interpret the temporal evolution of a gesture accurately. Many systems adopt a two-stage approach, first applying CNNs for frame-wise feature extraction, followed by LSTMs to model the sequential aspects of the gesture (Lee et al., 2021 [11]).

2.4.4 Integration of Recursive Neural Network Approaches

Recent work has also explored recursive neural network architectures for capturing hierarchical structures in sign language gestures (Borges-Galindo et al., 2024 [12]). Although our primary focus is on CNN-LSTM hybrids, recursive models offer a promising alternative for handling complex information in gestures and may be considered for future extensions of our system.

2.5 Base Repository for Implementation

Our ASL recognition system builds upon the open-source repository by K. Takahashi, “hand-gesture-recognition-mediapipe” [13]. This repository provides a robust baseline that leverages the MediaPipe library for real-time hand and body landmark detection. By integrating MediaPipe’s efficient tracking algorithms with our CNN-LSTM architecture, our system extracts key spatial features and processes them in real time for accurate gesture recognition.

This base repository has been instrumental in our progress, enabling rapid prototyping and rigorous testing under real-world conditions.

2.6 Challenges and Gaps in Current Systems

Despite significant advances in ASL recognition technologies, several critical challenges and gaps remain that hinder widespread adoption in real-world settings:

- **Limited Vocabulary and Generalization:** Many existing systems are designed to recognize only a restricted set of signs, making it difficult to scale to larger vocabularies. This limitation often stems from insufficient training data and models that are tuned for a narrow set of gestures (Debnath and Joe, 2024 [3]). Consequently, these systems struggle to generalize across different signers or adapt to variations in signing styles.

- **Difficulty Capturing Nonmanual Signals:** Effective sign language communication relies not only on hand gestures but also on nonmanual signals such as facial expressions and head tilts. Current approaches often neglect these critical features due to the complexities involved in accurately capturing and processing them. As a result, essential grammatical and emotional nuances are frequently lost, limiting the overall accuracy and naturalness of the interpretation (Avola et al., 2019 [4]).

- **Poor Performance in Real-World Environments:** Many systems demonstrate high accuracy under controlled conditions; however, performance typically degrades in real-world scenarios. Factors such as variable lighting conditions, dynamic backgrounds, and ambient noise can adversely affect both the feature extraction process and the robustness of the model (Liu et al., 2016 [6]). These challenges underscore the need for systems that maintain high performance in diverse and unpredictable environments.

- **User Interface and Accessibility Shortcomings:** Beyond the technical aspects of recognition, user experience plays a pivotal role in the adoption of ASL systems. Current implementations often suffer from complex calibration procedures, unintuitive interfaces, or hardware requirements that are not easily accessible to the Deaf community. Such issues can discourage user engagement and limit the practical utility of the technology (Hibbard et al., 2020 [1]).

2.7 Conclusion

In summary, while state-of-the-art ASL recognition systems have made commendable progress, significant gaps remain, particularly in vocabulary generalization, the integration of nonmanual signals, robustness in real-world conditions, and user accessibility. Our proposed system addresses these issues by leveraging a hybrid CNN-LSTM architecture integrated with MediaPipe for real-time, robust hand and body landmark detection. By focusing on a user-centered design and optimizing performance under varying environmental conditions, the system aims to offer:

- **Real-Time Performance:** Ensuring quick and accurate recognition through efficient deep learning models and streamlined data processing pipelines.
- **Robust Detection:** Enhancing accuracy across a wide range of signs, including both manual and nonmanual signals, by employing advanced feature extraction and temporal modeling techniques.
- **User-Friendly Interface:** Prioritizing accessibility and ease of use, with minimal calibration requirements and an intuitive design that caters to the needs of Deaf and hard-of-hearing individuals.

This conclusion sets the stage for the subsequent methodology chapter, where we will detail the system architecture, data acquisition procedures, and the experimental protocols implemented to validate our approach.

Chapter 3

Software and Application

This chapter describes the software structure, implementation, and integration of the components that form the ASL recognition system. The system incorporates computer vision techniques with deep learning techniques to enable real-time sign language recognition. Within this chapter, the research process is described, i.e., data collection, algorithms and techniques employed, evaluation metrics, user engagement, and limitations and future direction.

3.1 Research Approach

The entire process integrates computer vision technique with OpenCV and MediaPipe and a deep learning classifier in TensorFlow. Video frames are grabbed by an OpenCV webcam, which is then processed using MediaPipe's Hands solution to find the 21 hand landmarks (each of x, y, and z coordinates). These 63-dimensional feature vectors are fed into the neural network model.

MediaPipe hand landmark detection provides speed and resilience for real-time usages, according to the MediaPipe Hand Landmarker[2] and MediaPipe Hands GitHub documentation[3]. OpenCV 3.4 is used for taking the frames, getting the images ready, and placing the drawings on top of them for visualizing purposes, where each process stage is as efficient as effective[4].

Figure 2: System flowchart: from webcam input through OpenCV preprocessing, landmark extraction via MediaPipe, model inference with TensorFlow, to output display with overlays.

[2]`https://ai.google.dev/edge/mediapipe/solutions/vision/hand_landmarker`

[3]`https://github.com/google-ai-edge/mediapipe/blob/master/docs/solutions/hands.md`

[4]https://docs.opencv.org/3.4/

3.2 Data Collection

The primary training data are taken from the ASL Alphabet Dataset on Kaggle[5]. The dataset contains images categorized into letter class and contains additional classes of "space," "del," and "no_gesture" to simulate realistic text input conditions.

Horizontal flipping is employed to augment the dataset, introducing variability and allowing the model to generalize to different hand orientations. Literature on tackling data variability and augmentation techniques in sign language recognition (Papastratis et al., 2021 [9]; Ru and Sebastian, 2023 [10]) supports this approach.

3.3 Algorithms and Techniques

Landmark Extraction: MediaPipe Hands is set up in both the offline and real-time processing modules. In the real-time application (realized in `main_py`), every frame is resized to 640×480 for uniformity prior to BGR to RGB conversion. MediaPipe processes the image for detecting a hand and finding 21 landmarks. Every landmark gives three coordinates (x, y, and z), making it a 63-dimensional feature vector per frame.

```python
def extract_keypoints(image):
""" Extract hand keypoints from user using MediaPipe """
try:
    # Revise image -> better preformance
    image = cv.resize(image, (640, 480)) # (640, 480) is default webcam
        resolution
    # Consistent input size will help improve speed
    # and stability of landmark detection

    # Convert image to RGB spectrum
    image_rgb = cv.cvtColor(image, cv.COLOR_BGR2RGB) # OpenCV images are
         by default BGR format
    results = hands.process(image_rgb) # Process image with MediaPipe
```

[5]https://www.kaggle.com/datasets/grassknoted/asl-alphabet

```python

    if results.multi_hand_landmarks:
        landmarks = results.multi_hand_landmarks[0].landmark
        # Get hand's landmark
        # returns a list of 21 landmark points for a single hand
            detected
        # Each landmark has an x, y, and z coordinate
        keypoints = []
        for landmark in landmarks:
            keypoints.append(landmark.x)  # x-coord
            keypoints.append(landmark.y)  # Y-coord
            keypoints.append(landmark.z)  # Z-coord (depth)
        return keypoints
except Exception as e:
    print(f"Error processing image: {e}")
return None
```

Excerpt 1: main.py (Landmark Extraction)

Model Architecture: The baseline model, used in `train_model.py`, is a dense neural network with the 63-dimensional landmark vector as input. Its architecture consists of an input layer for the 63 features, followed by two hidden dense layers containing 128 and 64 neurons respectively. The ReLU activation function and dropout are used by both the hidden layers for regularization. The final layer is a softmax output layer which has 29 outputs: the characters A–Z plus additional outputs for "del," "space," and "no_gesture." In the future, it will experiment with the employment of LSTMs or Transformers to be able to more precisely model dynamic gestures so the system can better cope with short motion-based signs (e.g., "J" and "Z").

Real-Time Inference and Visualization: In the backend based on Flask (in `main.py`), video frames are grabbed in real time. Frames are processed to get hand landmarks and subsequently classified with the trained model (loaded from `asl_model.h5`). The system has a stability logic where a gesture detected needs to be consistent for a minimum of one second before it is taken as valid. When a stable gesture is identified, it is added to the recognized text, and feedback is given visually through overlays like bounding boxes, skeleton lines, and a flash effect inside the hand's bounding box.

```python
if detected_letter != "no_gesture":  # or remove this check if you want
    to show everything
current_time = time.time()
# If no stable letter, or if different letter detected, reset
if stable_letter is None or detected_letter != stable_letter:
    stable_letter = detected_letter
    stable_start_time = current_time
else:
    # Check if the letter has been stable for at least 1.00 seconds
    if current_time - stable_start_time >= 1.00:
        # Update recognized_text based on the stable detected gesture
        if stable_letter == "space":
            recognized_text += "␣"
        elif stable_letter == "del":
            recognized_text = recognized_text[:-1]
        else:
            recognized_text += stable_letter
        # Set flash start time for visual flash effect within the hand
            bounding box
        flash_start_time = current_time
        # Reset stable detection so it only triggers once per gesture
        stable_letter = None
        stable_start_time = None
```

Excerpt 2: generate.frames() in main_py (Gesture Stability Logic)

3.4 Evaluation Criteria

Model training is evaluated using standard metrics such as accuracy and loss, and confusion matrices are generated to identify commonly misclassified gestures. For real-time performance, frames per second (FPS) are monitored via a custom utility (`cvfpscalc.py`), targeting a rate of 15–30 FPS to ensure smooth interaction. *THIS WILL BE EXPANDED ON WHEN I GO INTO OTHER TRAINING MODELS I AM USING, WITH MULTIPLE SECTIONS, THIS WILL TRANSITION INTO RESULTS CHAPTER

Chapter 4

Results

Include text here

4.1

Include text here

Chapter 5

Conclusion

Include text here

5.1

Include text here

Bibliography

[1] E. Hibbard *et al.*, "Getting a Sign in Edgewise: User-Centered Design Considerations in Creating a Signed Language Mentoring Management System," *Sign Language Studies*, vol. 20, no. 2, pp. 264–300, 2020. Available: `https://www.jstor.org/stable/26983963`

[2] V. Falvo, L. P. Scatalon, and E. F. Barbosa, "The Role of Technology to Teaching and Learning Sign Languages: A Systematic Mapping," in *2020 IEEE Frontiers in Education Conference (FIE)*, Uppsala, Sweden, 2020, pp. 1–9, doi: `10.1109/FIE44824.2020.9274169`. Available: `https://ieeexplore-ieee-org.libezproxy2.syr.edu/document/9274169`

[3] J. Debnath and P. J. I R, "Real-Time Gesture Based Sign Language Recognition System," in *2024 International Conference on Advances in Data Engineering and Intelligent Computing Systems (ADICS)*, Chennai, India, 2024, pp. 01–06, doi: `10.1109/ADICS58448.2024.10533518`. Available: `https://ieeexplore-ieee-org.libezproxy2.syr.edu/document/10533518`

[4] D. Avola, M. Bernardi, L. Cinque, G. L. Foresti, and C. Massaroni, "Exploiting Recurrent Neural Networks and Leap Motion Controller for the Recognition of Sign Language and Semaphoric Hand Gestures," *IEEE Transactions on Multimedia*, vol. 21, no. 1, pp. 234–245, Jan. 2019, doi: `10.1109/TMM.2018.2856094`. Available: `https://ieeexplore.ieee.org/document/8410764`

[5] Stefanidis, Kiriakos, Dimitrios Konstantinidis, Thanassis Kalvourtzis, Kosmas Dimitropoulos, and Petros Daras. "3D technologies and applications in sign language." 2020. Available: `https://www.researchgate.net/publication/340966069_3D_technologies_and_applications_in_sign_language`

[6] T. Liu, W. Zhou, and H. Li, "Sign language recognition with long short-term memory," in *2016 IEEE International Conference on Image Processing (ICIP)*, Phoenix, AZ, USA, 2016, pp. 2871–2875, doi: `10.1109/ICIP.2016.7532884`. Available: `https://ieeexplore-ieee-org.libezproxy2.syr.edu/document/7532884`

[7] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," in *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, Nov. 15, 1997, doi: `10.1162/neco.1997.9.8.1735`. Available: `https://ieeexplore.ieee.org/document/6795963`

[8] C. Olah, "Understanding LSTM Networks," Understanding LSTM Networks, Aug. 17, 2015. [Online]. Available: `https://colah.github.io/posts/2015-08-Understanding-LSTMs/`

[9] I. Papastratis *et al.*, "Artificial Intelligence Technologies for Sign Language," *Sensors (Basel, Switzerland)*, vol. 21, no. 17, p. 5843, Aug. 30, 2021, doi: `10.3390/s21175843`. Available: `https://pmc.ncbi.nlm.nih.gov/articles/PMC8434597/`

[10] J. T. S. Ru and P. Sebastian, "Real-Time American Sign Language (ASL) Interpretation," in *2023 2nd International Conference on Vision Towards Emerging Trends in Communication and Networking Technologies (ViTECoN)*, Vellore, India, 2023, pp. 1–6, doi: `10.1109/ViTECoN58111.2023.10157157`. Available: `https://ieeexplore-ieee-org.libezproxy2.syr.edu/document/10157157`

[11] C. K. M. Lee, K. K. H. Ng, C.-H. Chen, H. C. W. Lau, S. Y. Chung, and T. Tsoi, "American sign language recognition and training method with recurrent neural network," *Expert Systems with Applications*, vol. 167, pp. 114403, 2021, ISSN 0957-4174. Available: `https://doi.org/10.1016/j.eswa.2020.114403`

[12] E. A. Borges-Galindo *et al.*, "Sign Language Interpreting System Using Recursive Neural Networks," *Applied Sciences*, vol. 14, no. 18, Art. no. 8560, Sep. 23, 2024, Available: `https://www.mdpi.com/2076-3417/14/18/8560`

[13] K. Takahashi, "hand-gesture-recognition-mediapipe," GitHub repository, Available: `https://github.com/kinivi/hand-gesture-recognition-mediapipe` (Accessed: Feb. 26, 2025)