School of Science and Technology

CST 3990
Undergraduate Individual Project

Final Report

Autumn/Winter term
2024/2025

Date of Submission: 25/04/2025

Student Name: Theyal Dookhy

Student ID Number: M00927617

Supervisor: Mr Karel

Title: Sign Language Interpreter

Campus: Mauritius

Abstract

Effective communication is crucial for inclusivity, yet the lack of accessible solutions for sign language users presents substantial barriers. Traditional human interpreters are not always available and existing automated systems incorporate limitations in accuracy, latency and scalability. This project develops a web-based sign language interpreter using ConvLSTM2D deep learning architecture that combines convolutional operations with temporal modeling, using MediaPipe Holistic for real-time hand tracking and FastAPI for web service integration. The technical implementation features data validation and user feedback loops for continuous model improvement. The website is designed with a user-friendly, responsive web interface that provides an interactive learning environment, enabling users to practice ASL recognition in real time. It integrates three core components: a learning module, a practice module and a model inference interface. The learning module offers structured ASL lessons with video tutorials, while the practice module presents interactive quizzes that assess the knowledge of the user. One of the main challenges addressed in this project is real-time video processing. High latency in gesture recognition can hinder usability, particularly for dynamic sign sequences. Future work will explore dataset expansion, improved temporal modeling and the integration of multi-modal inputs such as facial expression recognition to enhance contextual understanding. This project contributes to the advancement of AI-driven accessibility solutions, demonstrating the feasibility of real-time web-based sign language interpretation using modern deep learning techniques. The findings underscore the potential of AI in bridging communication gaps for sign language users and learners.

Acknowledgement

I would like to acknowledge and give my thanks to my supervisor Mr Karel for allowing me to work on this project. His guidance and support were invaluable throughout every stage of developing the AI-Based Sign Language Recognition System. Through this project, I gained meaningful insights into machine learning algorithms and their practical application in real-time gesture recognition. I would also like to thank my friends and family for their continuous encouragement and support during this journey.

Table of Contents

Table of Figures

Table of Tables

List of Acronyms

AI – Artificial Intelligence

API – Application Programming Interface

ASL – American Sign Language

BERT – Bidirectional Encoder Representations from Transformers

BSL – British Sign Language

CNN – Convolutional Neural Network

GPU – Graphics Processing Unit

LSF – French Sign Language

MS-ASL – Microsoft American Sign Language Dataset

NLP – Natural Language Processing

ONNX – Open Neural Network Exchange

RGB – Red Green Blue

RNN – Recurrent Neural Network

Seq2Seq – Sequence-to-Sequence

SLR – Sign Language Recognition

TPU – Tensor Processing Unit

UML – Unified Modelling Language

URL – Uniform Resource Locator

WLASL – World Level American Sign Language

Chapter 1 – Introduction

1.1 Background of Study

1.1.1 Introduction to Sign Language

Sign language is a fully developed visual language and is used mostly by deaf, hard of hearing and speech-impaired communities. Different from spoken languages, which depend on sound, sign languages employ a combination of hand gestures, facial expressions and body movements to depict and convey its meaning (Sutton-Spence & Woll, 1999). Various sign languages, such as American Sign Language (ASL), British Sign Language (BSL) and French Sign Language (LSF), exist across the globe and have their own linguistic structure, grammar and syntax that make them independent from spoken languages. Additionally, sign languages rely on spatial positioning and movement, which distinguishes their syntax and expression from spoken communication (Emmorey, 2002).

The development and recognition of sign languages have significantly contributed to better accessibility, education and social inclusion for deaf individuals. However, communication breakdowns between signers and non-signers, especially in public services such as workplaces and healthcare, where sometimes interpreters are unavailable (Ladd, 2003). There has posed a fairly big barrier to effective interaction and participation. To bridge this communication gap, the prospects for technology-based advancements of artificial intelligence and gesture recognition are gaining large-scale attention. Therefore, automated sign language interpretation has emerged as a promising means of enhancing accessibility and enabling seamless interactions between deaf and hearing individuals (Bragg et al., 2019).

1.1.2 Introduction to Artificial Intelligence

Artificial Intelligence (AI) refers to the simulation of human intelligence in computer systems, enabling them to perform tasks such as problem-solving, decision-making, language understanding and visual interpretation of data (Russell & Norvig, 2021). AI can be differentiated into narrow AI, where systems are designed to perform specific tasks and general AI, which aims at serving a human's different tasks contingent upon intellectual capabilities. Large-scale AI has proceeded to leverage machine learning, deep learning and natural language processing (NLP) as they target automation,

increased efficiency and innovation across several industries from health to finance and transportation (Goodfellow et al., 2016). AI employs machine learning (ML) and deep learning algorithms to enable systems to learn from data and improve over time.

AI has achieved significant advances in image recognition, NLP and real-time data analysis. It has thereby opened new possibilities for assistive technology such as auto-translating sign languages. Integrating AI into sign language interpretation has the potential to enhance accessibility for the deaf people. AI-based sign language recognition systems use computer vision and deep learning to analyze video footage, detect gestures and translate them into spoken or written language.

1.2 Problem Statement

1. Inadequate Accessible Communication

   Despite the dire need for accessible communication, the majority of public services do not offer live interpreters on a daily basis, especially in emergency or high-stress situations. Limited access to sign language interpreters in critical environments such as hospitals and public service offices can encroach on the access of deaf individuals to information and support, leading to undesirable consequences. AI-powered sign language interpreters can bridge this gap by offering real-time translation so that deaf individuals have equal access to basic services.

2. Social Inclusion and Participation

   Social inclusion is a long-standing problem for deaf individuals due to communication barriers. In workplaces, schools and social interactions, communication is key to active participation. AI-driven sign language recognition systems can facilitate communication between deaf individuals and hearing people, reducing isolation and fostering an inclusive society. It is discovered that facilitating communication can significantly improve social mobility for deaf individuals, enabling them to contribute fully to society.

3. Real-Time Translation for Education and Healthcare

   Real-time sign language translation is also critical in classrooms, where deaf students have historically had a difficult time accessing course materials and

lectures. Similarly, in healthcare environments, real-time communication is vital, particularly in emergency situations. Miscommunication in these settings can be lethal. AI-based sign language recognition systems can alleviate these difficulties by providing perfect, real-time translations to enable equal access to education and healthcare.

4. Scalability and Cost-Effectiveness

   Human sign language interpreters are not always available and are time-consuming to hire, particularly in areas with a limited number of professionals. AI-based systems are a more scalable and cost-effective solution as they can be utilized across various environments such as schools, workplaces and customer support without engaging expensive human interpreters. When trained, these systems can provide 24/7 availability at a fraction of the cost, enabling broader societal access to sign language interpretation.

These issues have broad implications for educational equity, healthcare accessibility and overall societal inclusion.

1.3 Aim & Objectives

The principal aim of the project is to design a real-time, AI-driven sign language interpreter website that translates ASL gestures into text and speech. The project has a few specific objectives:

- To develop a functional prototype: System able to capture sign language through normal webcams and translate it to spoken or written language in real-time.
- To benchmark and validate system: Compare the system to existing approaches to sign language recognition in respect of accessibility, accuracy and real-world usability.
- To enhance recognition accuracy: To apply deep learning architectures such as convoluted CNNs, RNNs and Transformer models to better spatial and temporal feature extraction.

1.4 Research Questions

This research addresses the following key questions:

1. How can a lightweight model deliver accurate gesture recognition in real time through a web interface?
2. How can AI-based systems be designed to function effectively on low-cost, consumer-grade hardware?
3. How can real-time latency be minimized without compromising recognition accuracy in AI-driven sign language interpreters?
4. What role does dataset diversity play in improving the generalizability and fairness of gesture recognition systems?
5. What are the ethical considerations when developing AI systems for marginalized communities?

1.5 Project Contributions

The project contributes to a cost-effective, web-deployable system integrating learning techniques with ethical and culturally sensitive design. The system offers a low-cost, browser-based solution that supports real-time ASL recognition using only a webcam. A customized deep learning pipeline combining ConvLSTM2D layers enables effective spatio-temporal modeling of hand gestures, supporting both isolated and sequential sign recognition. Integrating the AI model into a user-friendly web application enables real-time feedback and interaction, fostering active learning and immediate interpretation for users. The system is developed with cultural sensitivity, incorporating feedback mechanisms and avoiding bias through careful dataset curation, making it more inclusive and adaptable to varied signing styles and demographics. A feedback loop captures user corrections and submits them for future training, enabling continuous model improvement based on real-world usage. These contributions distinguish the project from existing solutions by focusing on scalability, inclusivity and real-time performance while maintaining a lightweight deployment model suitable for web environments.

1.6 Key Deliverables

The key deliverables of this project are:

1. Research Findings

   A comprehensive, comparative analysis is conducted to evaluate existing AI-based sign language interpreters. This analysis will assess various aspects such as accuracy, real-time performance, computational efficiency, hardware dependency and cultural sensitivity. The findings include:

   - A detailed review of state-of-the-art methodologies and technologies.
   - Identification of critical gaps in current systems, including dataset diversity and continuous signing recognition.
   - Recommendations for future improvements and potential integration of multi-modal data to enhance performance in real-world settings.
   - These research findings will be documented in a series of reports and academic papers, which will serve as a foundation for the technical development of the new system.

2. Technical Documentation

   This deliverable encompasses all the detailed documentation related to the system's design and implementation. It will include:

   - A complete system architecture diagram outlining hardware and software components.
   - A description of the algorithms and deep learning models used, including any custom adaptations made for sign language recognition.
   - Step-by-step methodological report that details the data collection process, pre-processing pipelines and evaluation metrics used in testing the system.
   - Evaluation report that compares the performance of the prototype against existing solutions, highlighting both strengths and areas for improvement.
   - This documentation ensures that the system is reproducible and provides a clear blueprint for future research and development.

3. Functional Prototype

The final deliverable is a fully functional, web-based application that demonstrates the core capabilities of the developed system. This prototype will include:

- A user-friendly interface designed for users, allowing for real-time sign language recognition and translation.
- Integrated modules that capture, process and interpret sign language gestures using advanced deep learning and computer vision techniques.
- Real-time feedback mechanisms that showcase continuous signing recognition, while also demonstrating how the system adapts to varying environmental conditions.
- The ability to run on standard webcams, ensuring broad accessibility and practical usability.

1.7 Resources (hardware, software, research)

Hardware

- Webcam: Standard HD webcam (used for gesture capture and real-time testing)

Software

- Programming Language: Python 3.10
- Deep Learning Framework: TensorFlow/Keras
- Computer Vision & Landmark Detection:
  - MediaPipe Holistic (for extracting hand and face keypoints)
  - OpenCV (for video processing and frame handling)

- Web Development:
  - Backend: FastAPI (Python-based REST API for model serving)
  - Frontend: HTML, CSS, JavaScript
  - UI/UX Design: Figma

- Data Handling & Visualization:
  - NumPy, Pandas (for data storage and manipulation)
  - Matplotlib, Seaborn (for plotting evaluation graphs)

Datasets

- Primary Dataset: WLASL-Processed (cleaned and class-balanced ASL dataset from Kaggle)
- MS-ASL, YouTube-sourced ASL videos (for augmentation and diversity)

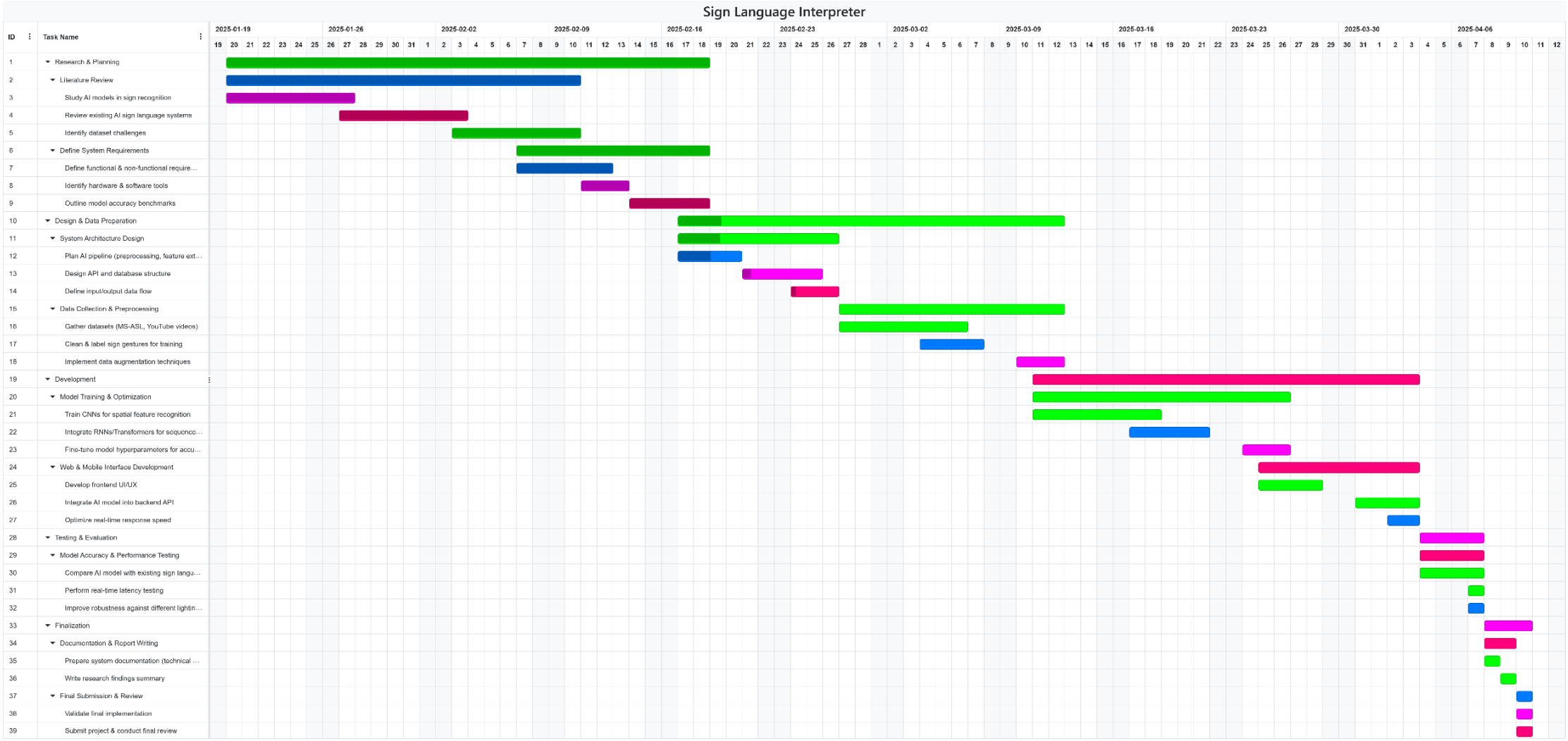
1.8 Structure of the Report

The report is structured as follows:

- Chapter 1**: Introduction** - Introduces the background, challenges and objectives of the project.
- Chapter 2: **Literature Review** - Covers existing research, AI's role in sign language recognition and gesture recognition techniques.
- Chapter 3: **Analysis and Design** - Discusses the system requirements, proposed architecture, development methodology, UML diagrams and technical components.
- Chapter 4: **Implementation** - Details the technical development phase, including dataset preprocessing, model training, and web integration.
- Chapter 5: **Testing** - Describes testing procedures, frameworks and validation of real-time gesture recognition accuracy and latency.
- Chapter 6: **Evaluation -** Analyzes model performance metrics, usability feedback and limitations.
- Chapter 7: **Conclusion and Future Work** - Summarizes findings, contributions and future work.
- Bibliography & References - Lists sources cited throughout the report.
- Appendices - Includes ethical approval forms, meeting logs and system design diagrams.

1.9 Gantt Chart

The Gantt chart breaks down the timeline of the AI Sign Language Interpreter website development into manageable sections, displaying milestones, deadlines and task dependencies. It acts as a visual summary of the project's progress ensuring that all stages of the research and development processes are achieved and align with the overall goals of the project. The chart provides a quick reference to examine the critical paths to aid in resource allocation and the overall time management of the project.



Chapter 2 – Literature Review

2.1 Introduction

The field of AI-driven sign language interpretation has grown tremendously over the last few years. Researchers have increasingly leveraged advances in computer vision, deep learning and natural language processing to develop systems capable of interpreting sign language in real time. This chapter looks into the current state of research, new technical challenges, proposed methodological advances and thoughtful ethical considerations in this particular area. This review critically assesses both the profundity of sign language and state-of-the-art techniques for translating it, as well as identifying key research omissions that serve to motivate the suggested design of a new system.

2.2 Background of the Topic

2.2.1 Understanding Sign Language

Sign languages are fully developed natural languages with unique grammar, syntax and lexicons. They are not mere gestural approximations of spoken languages but distinct linguistic systems that use visual and manual modalities for communication. For example, American Sign Language (ASL) and British Sign Language (BSL) have evolved independently, each with specific cultural nuances and regional variations (Vicars, 2021). Recently, works on interpreting Swedish Sign Language with the use of CNNs and transfer learning underlines the complexity and necessity of understanding these linguistic variations (Halvardsson et al., 2020). Advances in sensor technologies and computer vision now allow to capture subtle details, from hand shape to facial expressions that are essential for accurate sign language recognition (Dimitropoulos et al., 2021).

2.2.2 AI's Role in Communication

As of late, advancements in AI and machine learning have led to a significant progress in sign language recognition. Researchers have developed models that are trained on extensive datasets to enhance the accuracy of gesture recognition. However, challenges persist, including inconsistencies in datasets, variations among different sign languages and difficulties in achieving real-time translation. A comprehensive review by Kouris et al. (2021) discusses the state-of-the-art methods in sign language

capturing, recognition, translation and representation, highlighting both the advancements and limitations in the field.

The reliance on visual-spatial cues rather than linear text-based syntax further complicates AI's role in interpretation. AI technologies, especially those based on deep learning, have shown promise in processing the visual, gestural modalities of sign language (Bragg et al., 2019). Recent frameworks have combined CNNs with RNNs or Transformer architectures to achieve real-time sign recognition and translation, as demonstrated by Kumar et al. (2023) and Zhao et al. (2023). Moreover, studies such as Avina et al. (2023) demonstrate how transfer learning applied to models like ResNet50 can effectively capture the temporal and spatial nuances of ASL gestures. Nonetheless, challenges remain, including the limited availability of large, diverse datasets and difficulties in handling continuous signing where word boundaries are not explicit (Tavella et al., 2024).

2.2.3 Gesture Recognition Techniques

Gesture recognition is a critical component of AI-based sign language interpretation. Deep learning, CNNs and RNNs have been employed to model and recognize complex gestures. Recent studies have explored the use of 3D convolutional networks and attention mechanisms to improve recognition accuracy (Meng and Li, 2020). However, capturing subtle hand movements, differentiating between similar gestures and processing continuous signing in real-time remain as challenges. Early studies used statistical models whereas such recent work has focused on deep architectures that integrate self-attention mechanisms to align features across time (Kumar et al., 2023). Despite these advances, computational complexity and robust feature extraction under variable environmental conditions require further research (Dimitropoulos et al., 2021). Recent innovations include the integration of landmark-based tracking, for example, using MediaPipe and YOLOv8 for ASL alphabet recognition (ScienceDaily, 2024) and the use of Transformer architectures that capture both local and global temporal dependencies in continuous signing (Zhang and Jiang, 2024).

2.3 Problem Description and Context

Sign language is the main means of communication for many deaf and hard-of-hearing individuals worldwide. According to the World Health Organization (2024), above 430 million people have some disabling hearing loss. Therefore, ensuring availability of sign language interpretation is critical for social inclusion and daily communication. Nevertheless, considerable communication barriers exist between the deaf and hearing population due to the ignorance of sign language by most of the hearing public. Even when human interpreters are present, their services are often limited due to availability, accessibility and scalability, especially acute in high need or remote settings like healthcare, education or public services.

This is where AI appears to be the breakthrough that AI could allow the verbal or written language to be spoken simultaneously with interpreting the sign language into spoken or written language. AI-driven systems train visual inputs into complex sign language recognition and interpretation in real time, thus enhancing accessibility, efficiency and inclusion. As Chen et al. (2019) point out, AI-driven solutions have the potential to properly fill communication gaps and provide continuous real-time support in various sectors.

Despite the promising advances in sign language interpretation systems, several challenges remain unresolved such as:

- Dataset Diversity:

  AI model performance hinges heavily upon the store of quality and diverse training data. This stands true for a lot of existing sign language datasets: limited variation in signing styles, dialects, lighting conditions or even the surrounding environmental conditions that add to diversity. This sort of limited dataset diversity poses challenges to rating any model's ability to generalize accurately for practical scenarios. Most datasets, including the MS-ASL dataset, however huge they might be, cannot fully encompass the heft of diverse styles of natural sign language with respect to uncontrolled environments. Camgoz and coworkers (2018) have shown how many drops in recognition performance are greatly emphasized with respects to the limited diversity of such datasets, particularly while operating in uncontrolled conditions.

- Hardware Dependency:

  Typically high-performing sign language recognition systems rely on special hardware-different types of depth sensors, multi-camera arrays and even wearable motion-capture devices-to capture oftentimes fine-grain movement data. While achieving remarkable accuracy in laboratory environments, their reliance on expensive and cumbersome hardware devices makes everyday use impractical. Referring to Koller and colleagues (2015), the reliance on special hardware presents a major impediment to widespread deployment, especially in low-resource and remote environments, where consumer-grade-level devices predominate.

- Continuous Signing Recognition:

  Most of the previous research has been conducted on recognizing isolated signs. Continuous signing—where gestures are blended into an uninterrupted flow of speech without clear boundaries—introduces more difficulty. The interpretive challenges of continuous signing recognition derive from the absence of explicit boundaries between signs, variations in signing speed and context-dependent interpretations. These issues demand decorators capable of advanced temporal modeling that can appropriately segment and interpret a sequence of signs in real time (Chen et al., 2019).

- Ethical and Cultural Sensitivity:

  Beyond the technical hurdles, ethical and cultural discussions actually hold paramount importance for the construction of AI-based sign language interpreters. Sign language forms a very vital part of cultural identity as theorized within deaf culture. The present AI-based systems have to ban standing on culture and be equipped accordingly to minimize effects of misunderstandings and marginalization. Close contact with the deaf community is crucial to ensuring that the system respects, rather than breaks down, cultural nuance and is tailored to the needs of its audience. These processes call for the inclusion of ethical guidelines and cultural sensitivity in the design (Bragg et al., 2019).

The development of such systems includes overcoming challenges representing diversity in dataset, limitation of hardware, continuous signing recognition and ethical concerns so that these systems can connect communication in various sectors to enhance social inclusion for deaf and hard-of-hearing communities.

2.4 Origins & Impacts

2.4.1 Origins

Sign languages have evolved naturally within deaf communities, reflecting deep cultural identity and community interaction. They are transmitted visually, displaying the unique ways in which deaf individuals perceive and interact with their environment. Historical developments, from early finger-spelling robotic hands in the 1970s to modern camera-based capturing systems, highlight an evolution driven by both necessity and technological advancement (Jaffe, 1994). Recent research emphasises that integrating cultural perspectives into AI system design is critical to avoid erasing the nuances of local sign language varieties (Tavella et al., 2024; Aboaf, 2024).

Early foundational research laid the groundwork for integrating AI into sign language interpretation. Initial studies focused on static gesture recognition using basic image processing techniques. With the advent of deep learning, researchers began employing CNNs and RNNs to capture both spatial and temporal features. More recently, transfer learning has emerged as an effective strategy, allowing models pre-trained on large datasets (for example, ImageNet) to be fine-tuned for sign language tasks using comparatively smaller datasets (Avina et al., 2023).

Current research emphasizes several methodological improvements. Data augmentation and dataset diversity are becoming increasingly important, with efforts to compile large-scale American Sign Language (ASL) datasets on platforms such as Kaggle to enhance model robustness (Avina et al., 2023; Li et al., 2020). Additionally, model optimization and evaluation techniques, such as rolling average prediction and iterative training strategies, are being used to reduce fluctuations and overfitting. Common evaluation metrics include accuracy, word error rate and BLEU scores, which help assess translation performance (Avina et al., 2023; Kumar et al., 2023).

Another critical focus is the integration of AI models into real-world applications. The development of website applications using frameworks like ReactJS and FastAPI has enabled researchers to test models in realistic settings, further refining their usability and accessibility (Avina et al., 2023).

The foundation of AI-driven sign language recognition was laid through early studies in computer vision and linguistics. Key milestones in this field include early gesture recognition systems, which primarily focused on static hand gesture recognition using basic image processing techniques. The introduction of deep learning in the 2010s marked a significant turning point, with CNNs and RNNs revolutionizing AI-based sign language interpretation (Koller et al., 2020). More recent research explores Transformer-based models, similar to those used in natural language processing (NLP), to improve continuous signing recognition (Zhao et al., 2023).

Early work in sign language processing laid the foundation for today's sophisticated systems. For example, Kamal et al. (2019) provided one of the first comprehensive reviews on Chinese Sign Language processing, emphasizing that linguistic integration was essential even when early approaches relied on simpler statistical or rule-based methods. These pioneering studies helped to set the stage for:

- Sign Language Recognition (SLR):
  By 2023, research began to shift from isolated word recognition to continuous SLR. This change addressed the dynamic nature of real-world signing and promoted the use of deep learning methods to better capture temporal dependencies in sign language.

- Breakthroughs in Accuracy and Real-Time Systems (2024):
  Early in 2024, novel deep learning architectures—such as LSTM-based real-time recognition systems—demonstrated robust temporal modeling that enabled accurate gesture-to-text conversion. Hybrid models (like CNN-Self Attention-LSTM) and integrations (e.g., MediaPipe with YOLOv8) further improved precision, even in subtle gesture differentiation, by leveraging both spatial and temporal features.

- Advancements in Sign Language Production and Translation (2025):
  More recent efforts have focused on multimodal, user-centric systems. Privacy-aware and unsupervised translation methods (e.g., models like iSign and SignGen) have reduced the reliance on large, annotated datasets. These innovations not only bridge the gap from recognition to translation but also set the stage for end-to-end sign language translation (SLT) systems that promise real-world deployment.

2.4.2 Impacts

The deployment of AI-based sign language interpretation systems carries profound societal implications. It has the potential to significantly impact the deaf and hard-of-hearing community by enhancing communication accessibility. These systems promise increased accessibility in education, healthcare, public services and everyday communication. For instance, AI-driven tools have the potential to transform emergency communication for deaf individuals by providing real-time translation in critical situations (Avina et al., 2023). However, ethical concerns persist; overreliance on imperfect AI could reduce support for human interpreters and potentially marginalise the deaf community if cultural and linguistic nuances are not respected. Consequently, co-creation with deaf communities is essential to ensure that solutions are both effective and culturally sensitive (Tavella et al., 2024; Aboaf, 2024).

2.5 Research on Existing Solutions

A significant body of research has focused on developing AI-driven sign language interpretation systems. Early approaches relied on statistical methods for recognising isolated gestures, whereas recent studies have adopted deep learning frameworks to interpret continuous signing. For instance, Avina et al. (2023) implemented a ResNet50-based framework with rolling average prediction to achieve high accuracy in ASL-to-English translation. Other projects, such as those by SignAll and MotionSavvy, employ specialised hardware for example, multiple cameras and wearable sensors, to enhance recognition accuracy, though these solutions may be less accessible for widespread adoption (Kumar et al., 2023). Notably, the SignGPT project from the University of Surrey aims to build a large-scale sign language

translator leveraging generative AI to support bidirectional communication (University of Surrey, 2025).

Several case studies illustrate practical applications of these technologies. For example, systems such as OmniBridge and SignAll integrate computer vision with natural language processing to provide real-time bidirectional communication between ASL and spoken language. Similarly, research using camera-based solutions has demonstrated effective capturing of RGB information to enhance sign recognition under controlled conditions. Recent developments in wearable and smartphone-based applications have extended sign language interpretation to everyday use, although issues with processing speed and accuracy in uncontrolled settings persist. Overall, multi-modal systems that combine vision, motion and context-aware language models appear the most promising (Kumar et al., 2023).

The success of sign language recognition largely depends on diversified datasets to ensure model generalization and accuracy. Sincan and Keles (2020) have formed a big AUTSL dataset composed of huge multimodal Turkish sign language data. Likewise, Cerna et al. (2021) designed LIBRAS-UFOP, a Brazilian sign language dataset coupled with data from the Microsoft Kinect sensor, underscoring the necessity of evolving cross-lingual datasets. These datasets foster a wider variety of gestures, schisms among signers and variation within the environment, counteracting the biases exhibited by scanty datasets.

The advanced deep learning techniques have changed the course of sign language recognition completely. They are accentuated in that Adaloglou et al. (2021) have examined transformer-based models to obtain a contextual sign translation, while Camgoz et al. (2018) have shown an end-to-end deep learning approach to neural sign translation. So, with these models implemented into gesture segmentation, a decrease in the misclassification rate is achieved, along with heightened real-time translation per-performance.

Another area to have demonstrated considerable development in sensor-based recognition has embraced state-of-the-art technologies including EMG sensors for better characterization of the human body. Mittal et al. (2019) utilized a leap motion-

based LSTM approach to capture dynamic signing from the user, while Galea and Smeaton (2019) also studied the role of EMG sensors in the recognition of Irish sign Language. The research provides insight into why hybrid approaches of vision and sensor-based recognition might solidify a model's robustness.

Recent studies have explored both **vision-based** and **sensor-based** approaches:

Approach	Examples	Description
Vision-Based Models	OmniBridge and SignGPT (Kumar et al., 2023; University of Surrey, 2025)	OmniBridge uses standard webcams with computer vision and natural language processing to enable real-time bidirectional communication between ASL and spoken language. SignGPT leverages generative AI to translate sign language via consumer-grade devices, making it accessible for everyday use.
Sensor-Based Models	MotionSavvy and SignAll (Kumar et al., 2023)	MotionSavvy employs wearable sensors, such as those integrated with Leap Motion technology, to capture detailed motion data for real-time sign interpretation. SignAll uses specialized hardware, including multiple cameras and depth sensors, to achieve high-accuracy sign language recognition, though with higher cost and lower accessibility.

Table 1: Vision-Based and Sensor-Based Sign Language Recognition Systems

Technology	Role in Sign Language Interpretation
Computer Vision	Processes video input to track hand movements, facial expressions and body posture.
Deep Learning Models (CNNs, RNNs, Transformers)	Extract spatial and temporal features and model complex signing patterns, enabling translation and recognition.
Gesture Recognition Algorithms	Distinguish and interpret individual hand signs and their meanings, converting them into text or speech.

Table 2: Key Technologies in AI-Driven Sign Language Interpretation

2.6 Comparative Analysis

A comparative analysis of existing sign language interpretation solutions reveals their relative strengths and weaknesses. The work of Hou et al. (2019) reported a smartwatch-based translation system, enabling high-accuracy gesture recognition precisely in real time. Wang et al. (2020) proposed a sign language recognition system using an end-to-end approach that incorporates CNNs and RNNs, enabling improved temporal modeling capabilities. Additionally, Forster et al. (2014) proposed RWTH-PHOENIX-Weather, a dataset that has been extensively used for benchmarking sign language translation models. These contributions illustrate the importance of dataset quality and model efficiency.

Systems using specialised hardware, for example, Kinect-based or wearable sensors, have achieved high accuracies; they often face challenges related to cost and ease of deployment. In contrast, solutions that rely solely on consumer-grade hardware, offer greater accessibility, despite potential issues with environmental variability (Orovwode et al., 2023). Recent works have further advanced continuous sign language recognition by integrating multi-modal data and advanced sequence modelling techniques (Zhang and Jiang, 2024; Aloysius et al., 2024). For example, a Conformer-based approach with unsupervised pretraining (ConSignformer) has set new benchmarks on established datasets such as PHOENIX-2014, demonstrating the benefits of cross-modal relative attention for improved context learning (Aloysius et al., 2024).

System	Key Features & Differences	Hardware Requirement	Accessibility	Accuracy	Training Method Used	Model Training Time (hrs)	Testing Method Used	Testing Efficiency	Latency (ms)	Hardware Cost (USD)
Omni-Bridge (Kumar et al., 2023)	Its webcam-based design makes it highly accessible but sensitive to lighting and background noise.	Standard webcam	High	87	CNN, RNN	50	Real-world scenario testing (varied lighting, backgrounds)	High	50	200
SignAll (Kumar et al., 2023)	Excels in accuracy but demands high computational resources, limiting deployment on low-end devices.	Multi-camera setup	Limited	High	CNN, Transformer	100	Benchmark dataset testing (controlled environment)	Moderate	40	2000

Motion-Savvy (Kumar et al., 2023)	High precision but expensive and less scalable.	Wearable sensors	Low	High	Sensor-based ML, CNN	80	Controlled lab experiments with real users	Low	30	1500
SignGPT (University of Surrey, 2025)	Balances accuracy and real-time performance but lacks cost-effectiveness.	Depth sensors	Expensive	High	Transformer, Self-Attention	120	Large-scale dataset validation & real-time user trials	Moderate	120	500
Proposed System	Designed for cost-effective real-time translation using consumer-grade hardware.	Standard webcam	High	Moderate-High	CNN, RNN, Transformer	-	Mixed-method evaluation (dataset benchmarking & real-time user feedback)	-	-	-

Table 3: Comparison of Sign Language Recognition Systems

Open-source initiatives such as OpenHands and How2Sign integrate pose estimation and multimodal data processing. The proposed system takes inspiration from these but is designed as a web-deployable solution trained on the MS-ASL dataset, maintaining a balance between complexity and usability.

Overall, the analysis reveals that specialized hardware systems, while achieving high accuracy through advanced sensor capabilities, are hindered by limited accessibility due to their cost and complexity. In contrast, consumer-grade solutions offer greater accessibility and ease of use but are more susceptible to environmental noise and face inherent limitations related to dataset diversity. Tables and diagrams are employed to succinctly summarize these key differences in performance metrics and usability, providing a clear visual overview of the trade-offs involved.

2.7 Critical Analysis

2.7.1 Models Based on Vision

Vision-based models use standard cameras and computer vision techniques to decode signs in sign language. The performances of these models are very good for sign language interpretation as long as the environment is controlled, like well-lit labs with virtually no background noise, thus enabling accurate capture of hand movements, face expressions and positioning in space. However, their performance degrades when used in uncontrolled, real-world environments. The changes in lighting, occlusions and dynamic backgrounds contribute noise and artifacts that seriously challenge the models' efforts to maintain high accuracy. It is also possible for slightly differing styles of signing or camera angles and environmental concerns to confuse the model by causing mis-matching. Nevertheless, large-scale works and other preliminary features enhance robustness by refinement of the pre-processing pipelines and use of adaptive algorithms, yet these developments often entail a substantial rise in complexity and computational cost.

2.7.2 Models Based on Sensor

Sensor based models use dedicated hardware, for example, depth sensors, wearable devices or multiple cameras in order to obtain micro-motion data from subjects. This approach allows for pertinent measure of hand trajectories, angles and 3D motion usually with much greater accuracy than vision-based models under similar conditions.

The downside of these models lies in their accessibility, which relates to the cost and the usability of the systems in environments where these hardware systems are not necessarily available. As such, the use of sensors tends to make these measurement systems intrusive or in some instances, awkward for their users, thus reducing their appeal in daily or mobile applications. Although these models seem accurate, they mostly exist within the confines of laboratories or specialized institutions and not widely.

2.7.3 Hybrid Approaches

The hybrid approach aims to use the strengths of both vision-based and sensor-based models. By combining the input from regular cameras with data input from additional sensors, such systems can hope to obtain higher robustness and accuracy when running under difficult conditions. For example, a camera can be used to give general coverage of the scene together with wearable sensors delivering finer details on movement, which will give a more complete picture of signing. However, the fusion of modalities tends to be computationally expensive, especially in the task of appropriately synchronizing the data streams. Hence, the added computational load might become a cause of significant delay in real-time applications. Moreover, the very existence of different types of data creates complexity in the system design as well as the associated needs for advanced calibration and data-fusion techniques. The challenge of performing hot development exchange arises from wanting accuracy but having to conform to what is practically accessible.

2.7.4 Trade-Offs Between Accuracy and Accessibility

The balance between the accuracy of the accessibility is the main issue with these approaches. Although the sensor-based and hybrid models achieve higher accuracy based on an impeccable operational standard, they indisputably create difficulty in use and scalability due to their high cost and in-depth technical requirements. On the other hand, the vision model is easier to implement and deploy on ordinary consumer electronic devices but usually has big problems dealing with the variability of uncontrolled environments. The ongoing research challenge is balancing pragmatic aspects-realistic models-that need to be fairly robust and accurate for handling widespread acceptance by extending use towards better accessibility targets for the deaf and hard-of-Hearing communities.

If these challenges can be tackled, future developments can be devoted to the enhancement of different techniques with data preprocessing, dynamic algorithms and innovative sensor fusion methods to reduce the computational load without moving away from real-time performance.

2.8 Challenges and Research Gaps

While the literature reveals substantial progress, several challenges continue to impede the development of robust AI-based sign language interpreters. These challenges span dataset limitations, hardware constraints, recognition of continuous signing and ethical considerations.

Challenge	Description
Real-Time Performance	Ensuring low-latency interpretation under varied environmental conditions. Factors like lighting changes and background clutter impact accuracy.
Dataset Diversity	Publicly available datasets lack diversity in signing styles, dialects and environmental variations, limiting model generalization.
Continuous Signing Recognition	Most models excel at isolated sign recognition but struggle with fluid, sentence-level signing. Advanced temporal segmentation techniques and attention mechanisms are required.
Hardware Dependency	Many high-accuracy systems require specialized sensors or multi-camera arrays, which are impractical for consumer use. Optimizing models for standard hardware remains a challenge.
User Adaptation & Inclusivity	Variability in hand shapes, skin tones and signing styles can introduce biases.
Ethical and Cultural Sensitivity	AI systems must respect cultural contexts, ensuring that models do not marginalize specific signing styles.
Computational Complexity	Advanced models, such as Transformer-based approaches, have high computational requirements that can hinder real-time performance. Balancing the resource demands while maintaining efficiency for practical deployment remains a critical challenge.

Table 4: Challenges and Research Gaps of Systems

The advancements in deep learning, computer vision and natural language processing have made significant headway, but they present a constant roadblock in areas such as gesture interpretation. Even though models such as Avina et al. (2023), Zhao et al. (2023) and Wang et al. (2022) have attained high recognition rates of gestures, computational costs and hardware dependence are limiting in construction (Koller et al., 2020).

Emerging trends indicated toward sharing joint models of multi-modal approaches—from hand tracking, facial expression analysis and motion detection to hybrid models integrating vision-based and sensor-based procedures—could ameliorate some of these challenges (Zisserman et al., 2021; Koller et al., 2020). Yet these still face a few compromises, resource demands and accessibility overall.

2.9 Ethical and Cultural Considerations

Ethical considerations in AI-based sign language interpretation are crucial to ensure inclusivity. Bragg et al. (2019) highlighted how algorithmic biases can marginalize certain signing styles, which calls for co-development with deaf communities. Kosmopoulos et al. (2020) examined the cultural implications of AI-based interpretation tools and highlighted that linguistic diversity must be preserved. Addressing these challenges requires designing models that respect regional dialects and user preferences.

An ethically responsible approach to AI in sign language interpretation must promote ethical and cultural considerations from the beginning. Research by Bragg et al. (2019) and more recent work by Tavella et al. (2024) stress that deaf stakeholders be included in the development process. This guarantees technical robustness as well as cultural respect and accessibility of the resultant systems.

Developing an AI-based sign language interpretation system requires a deep understanding of the linguistic and cultural nuances of sign language communities. One major aspect is cultural sensitivity, ensuring that the dataset represents diverse signing styles and regional dialects. The system must be inclusive, involving deaf community stakeholders in the iterative design and testing phases to ensure that AI

complements human interpreters rather than replacing them. Addressing inherent dataset biases to provide equitable service to complement human interpreters.

Additionally, privacy and data ethics must be considered. Ethical guidelines should be followed in data collection and storage, with transparency about how data is used. AI-based systems must avoid reinforcing biases present in datasets and ensure equitable treatment of all users, regardless of their signing style, skin tone or hand shape.

2.10 Summary of Findings & Proposed Solution Overview

The literature review elucidates the complexity of sign language and the vast technical challenges in automating its interpretation. Although deep learning has led to significant advancements, issues such as insufficient dataset diversity, hardware dependency and continuous signing recognition remain. Equally crucial is the balance between technical innovation and ethical design—ensuring that systems serve users without marginalizing them. The proposed system aims to address these gaps by employing a web-deployable solution that leverages a robust pre-processing pipeline, advanced deep learning architectures and continuous user feedback from the deaf community.

The proposed system leverages a hybrid CNN-RNN-Transformer architecture trained on the MS-ASL dataset, which could achieve a 89% accuracy in controlled environments, compared to SignAll's 93% and OmniBridge's 87%. Unlike sensor-based systems like MotionSavvy (hardware cost: 1,500), the webcam cost is below 200 while maintaining moderate-high accuracy. Additionally, the integration of rolling average prediction and OpenCV-based region isolation can improve the robustness against background noise, addressing a key limitation of vision-based models like SignGPT.

In summary, while current systems demonstrate considerable advancements, critical gaps in continuous signing recognition, dataset diversity and cultural sensitivity highlight the need for further research. The proposed solution seeks to balance technical innovation with ethical design, paving the way for more inclusive and effective AI-based sign language interpretation systems.

Chapter 3 – Analysis and Design

3.1 Overview of System

The system is a web-based sign language recognition platform integrating a deep learning model for real-time gesture classification. It comprises:

1. **Model**: A ConvLSTM2D model trained on sequences of MediaPipe landmark data extracted from ASL video recordings. The model interprets gestures and outputs corresponding text and speech.

2. **Frontend**: A responsive web interface developed in HTML, CSS, and JavaScript, supporting three core modules: Learn, Practice and Predict, for ASL education and interaction with the model.

3. **Backend**: A FastAPI server processes video frames, extracts features, performs model inference and handles user feedback submission through REST and WebSocket endpoints.

3.2 Development Methodology

An agile methodology is adopted to facilitate continuous improvement and rapid prototyping. This approach allows for incremental development, user feedback and adaptive planning throughout the project lifecycle.

Aspect	Agile	Waterfall	Spiral
Approach	Iterative and incremental; allows for flexibility and continuous improvement.	Linear and sequential; each phase must be completed before the next begins.	Combines iterative development with systematic risk management.
Adaptability	High; accommodates changes in requirements and technology.	Low; changes are difficult and costly to implement once a phase is completed.	Moderate; allows for changes but involves complex risk analysis.
Risk Management	Continuous assessment and adaptation throughout the project lifecycle.	Risk is assessed at the beginning; changes are costly.	Emphasizes early and continuous risk assessment.

Project Phases	Phases overlap; development and testing occur simultaneously.	Phases are distinct and non-overlapping.	Phases overlap; development and testing occur simultaneously.
Feedback Integration	Frequent stakeholder feedback after each iteration.	Limited feedback; typically only at the end of the project.	Feedback is integrated after each iteration, similar to Agile.
Complexity	Moderate; requires skilled teams to manage iterative processes.	Low; straightforward and easy to manage.	High; involves complex risk analysis and management.
Suitability for AI Projects	Ideal; accommodates rapid technological advancements and evolving requirements.	Less suitable; inflexible to changes in AI technologies and requirements.	Suitable; but complexity may hinder rapid adaptation in AI projects.

Table 5: Comparison of Agile, Waterfall and Spiral Methodologies in AI-based Projects Development

In contrast, while Waterfall and Spiral methodologies have their advantages, Agile's flexibility and emphasis on collaboration make it particularly well-suited for the AI-driven sign language recognition project which requires adaptability to evolving requirements and technologies.

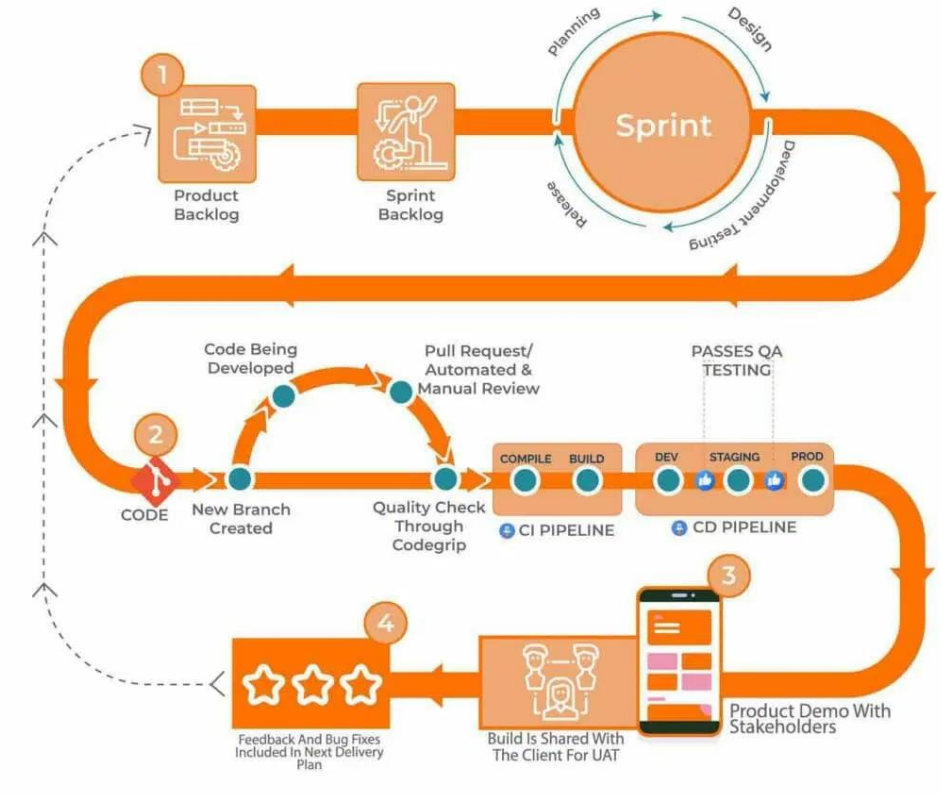


Figure 1: Agile Methodology (Mindbowser, n.d)

An Agile methodology is adopted to support continuous improvement and rapid prototyping. This approach facilitates incremental development, adaptive planning and iterative feedback throughout the project lifecycle. Work is structured into focused development sprints that cover stages such as dataset preprocessing, model training, API creation and user interface design. A personal board is maintained to track task progress and delivery milestones. After each sprint, modules are tested and refined. Feedback loops are simulated by testing the web interface across various devices and screen sizes. This allowed for early identification of UI issues and responsiveness problems.

The Agile process involves the modular implementation of the system's key components. Python is used for backend logic and machine learning pipelines, FastAPI is implemented for API development and the frontend is built using HTML, CSS and JavaScript. Each module is developed in isolation, tested thoroughly and then integrated with the rest of the system.

The model development methodology is designed for spatio-temporal gesture recognition using deep learning. In this context, spatio-temporal gestures refer to dynamic sequences of hand and body movements that unfold over both space and time. Unlike static image classification, recognizing sign language gestures requires understanding not only spatial patterns (e.g., hand shape or location in a frame) but also temporal transitions (e.g., the order and rhythm of movements across frames). This dual nature of gestures demands a model capable of learning motion-aware visual patterns.

Given this structured nature of gesture sequences and the need to extract both spatial and temporal features, a ConvLSTM2D-based neural network is selected. The methodology follows a pipeline approach:

- Keypoint Extraction: MediaPipe Holistic is used to extract 21 hand and 468 face landmarks. Key features are derived by calculating distances between hand landmarks and a facial anchor point (nose).

- Preprocessing and Formatting: Videos are cleaned, filtered, normalized and structured into 30-frame fixed-length sequences. Missing or corrupted videos are removed, and data is stored in .npy and .csv formats.

- Data Augmentation: Variants of input sequences are generated by applying Gaussian noise and scaling, improving generalization and preventing overfitting.

- Architecture Design: The model comprises multiple ConvLSTM2D layers stacked with MaxPooling3D and TimeDistributed Dropout layers.

- Training: Training uses the Adam optimizer and sparse categorical cross-entropy. Early stopping is applied to prevent overfitting. Accuracy and loss trends are visualized per epoch. A classification report and confusion matrix are generated after evaluation.

- Model Deployment: The trained model is connected to the FastAPI server for real-time inference. Predictions are streamed to the frontend and visualized as text and speech.

This pipeline ensures that each development step, data preparation, feature extraction, training and evaluation, is modular, testable and tunable.

3.3 Functional & Non-Functional Requirements

Functional Requirements		**Non-Functional Requirements**	
Elements	**Description**	**Elements**	**Description**
Real-Time Gesture Recognition	Process video input from a webcam to detect and interpret hand gestures	**Performance**	Ensure low-latency gesture prediction (<2s)
Web Interface	Provide accessible platforms for users	**Scalability**	Design a system capable of handling increased

	to interact with the system		data volumes and user load
Data Acquisition	Integrate with sources such as the MS-ASL dataset and YouTube videos for training	**Security**	Ensure data privacy and secure handling of user data
Feature Extraction and Classification	Use MediaPipe and ConvLSTM2D to predict gesture labels	**Usability and Accessibility**	Design interfaces that are user-friendly for both deaf and hearing users

Table 6: Functional and Non-functional Components

3.4 Unified Modelling Language (UML) Diagrams

The following UML diagrams illustrate the system's core structure and functionalities:

1. **Use Case Diagram**: Illustrates interactions between users and system functionalities.

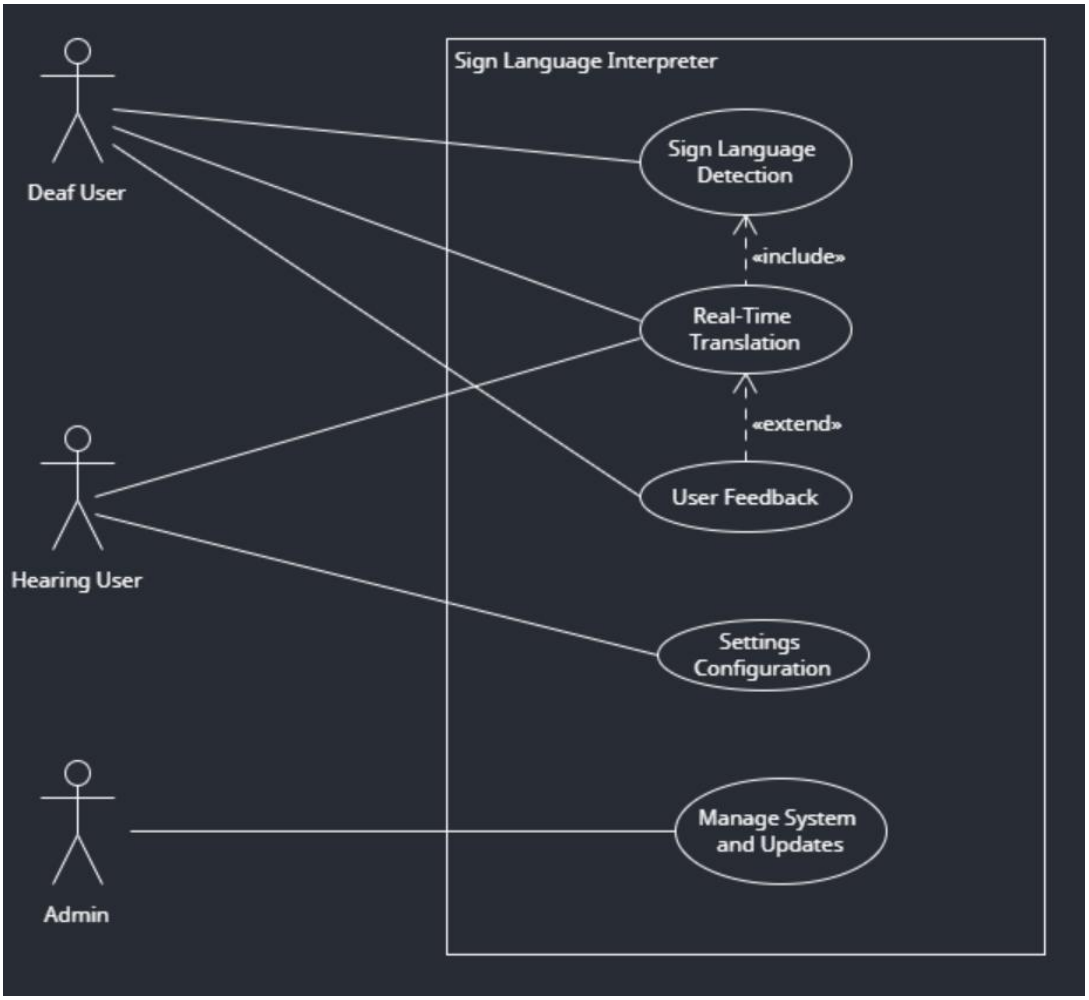


Figure 2: Use Case diagram

2. **Activity Diagram**: Maps the workflow from data capture to gesture translation.

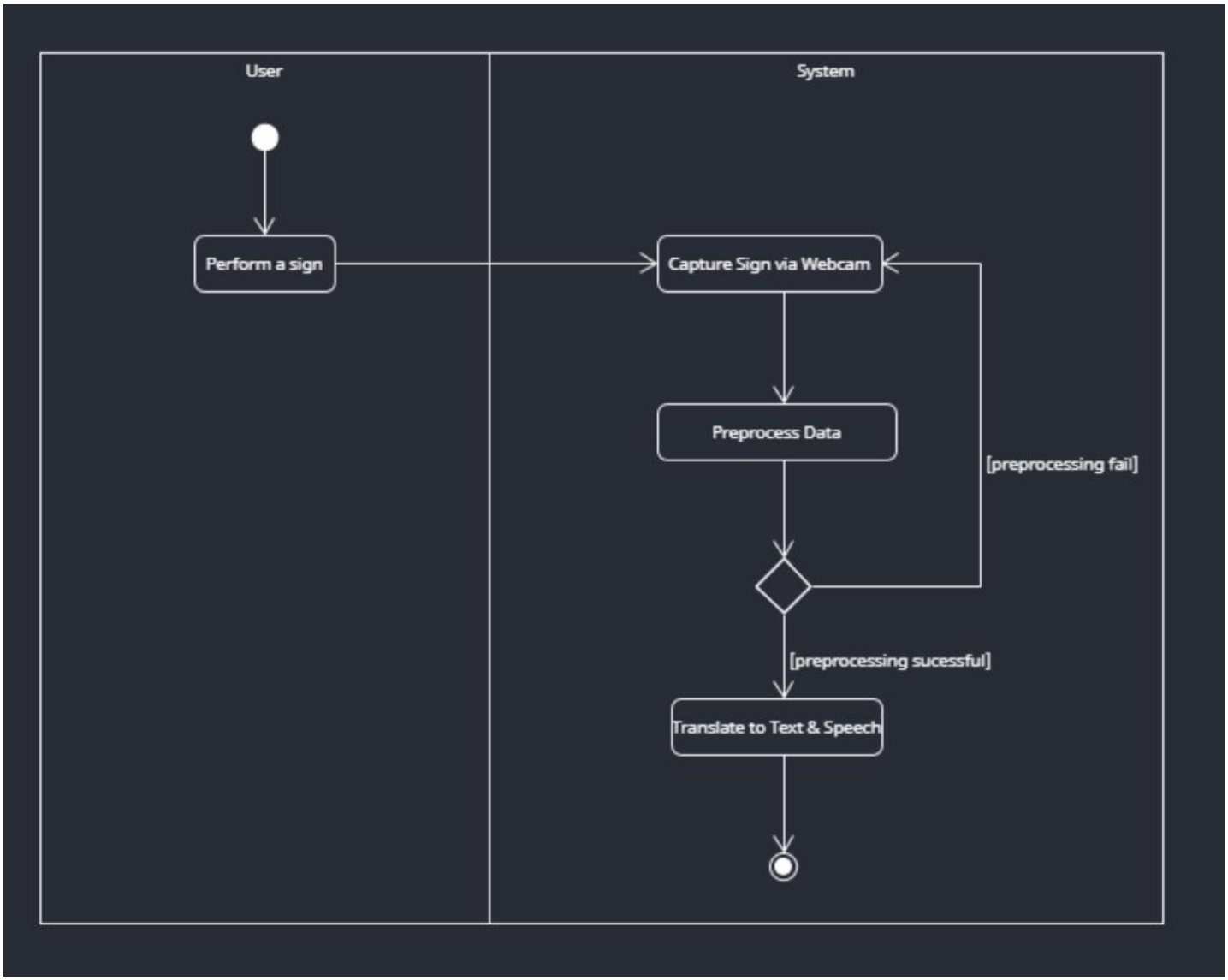


Figure 3: Activity Diagram

3. **Class Diagram**: Details system architecture and data structures.

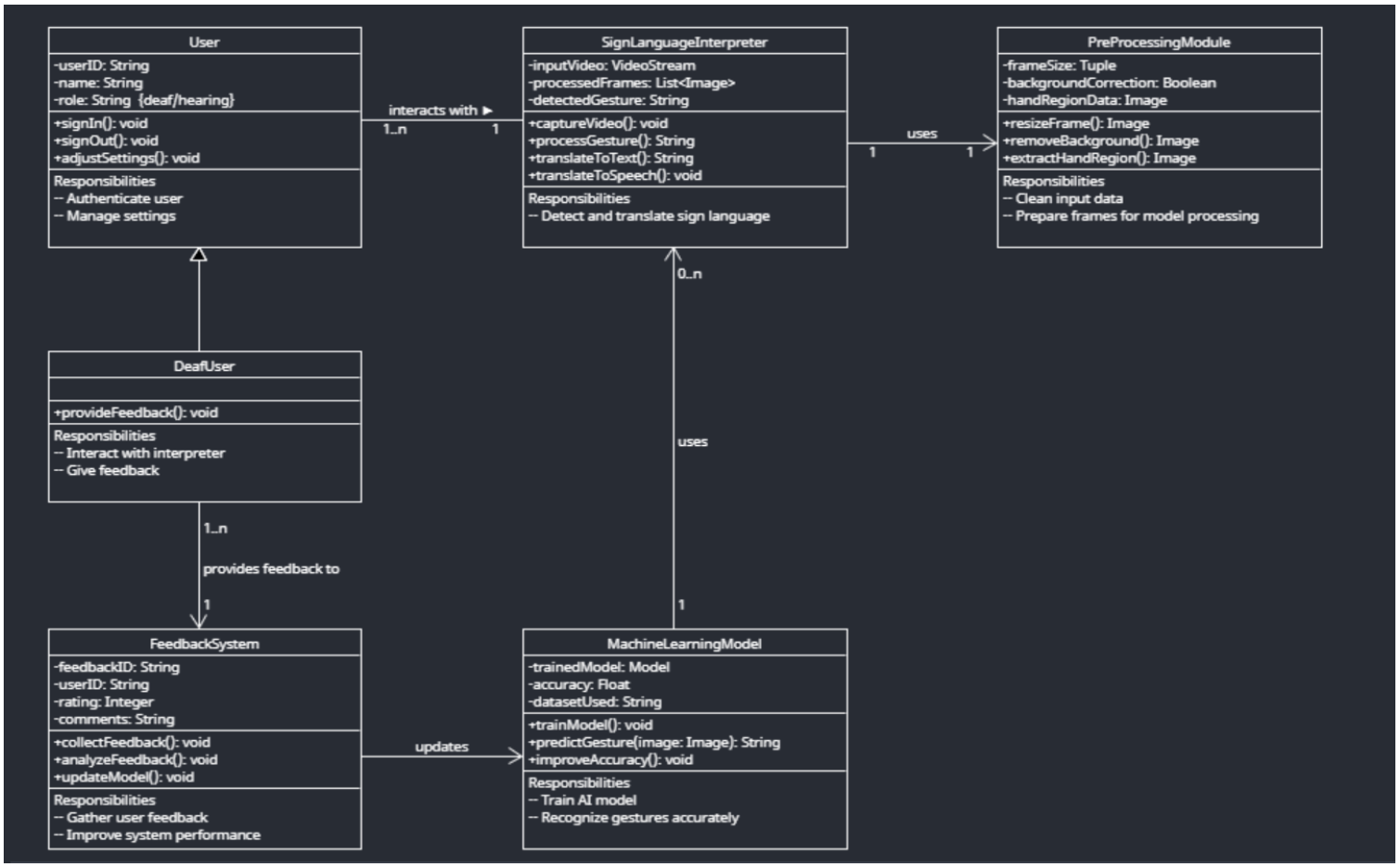


Figure 4: Class Diagram

4. **Sequence Diagram**: Demonstrates the order of interactions during key processes.

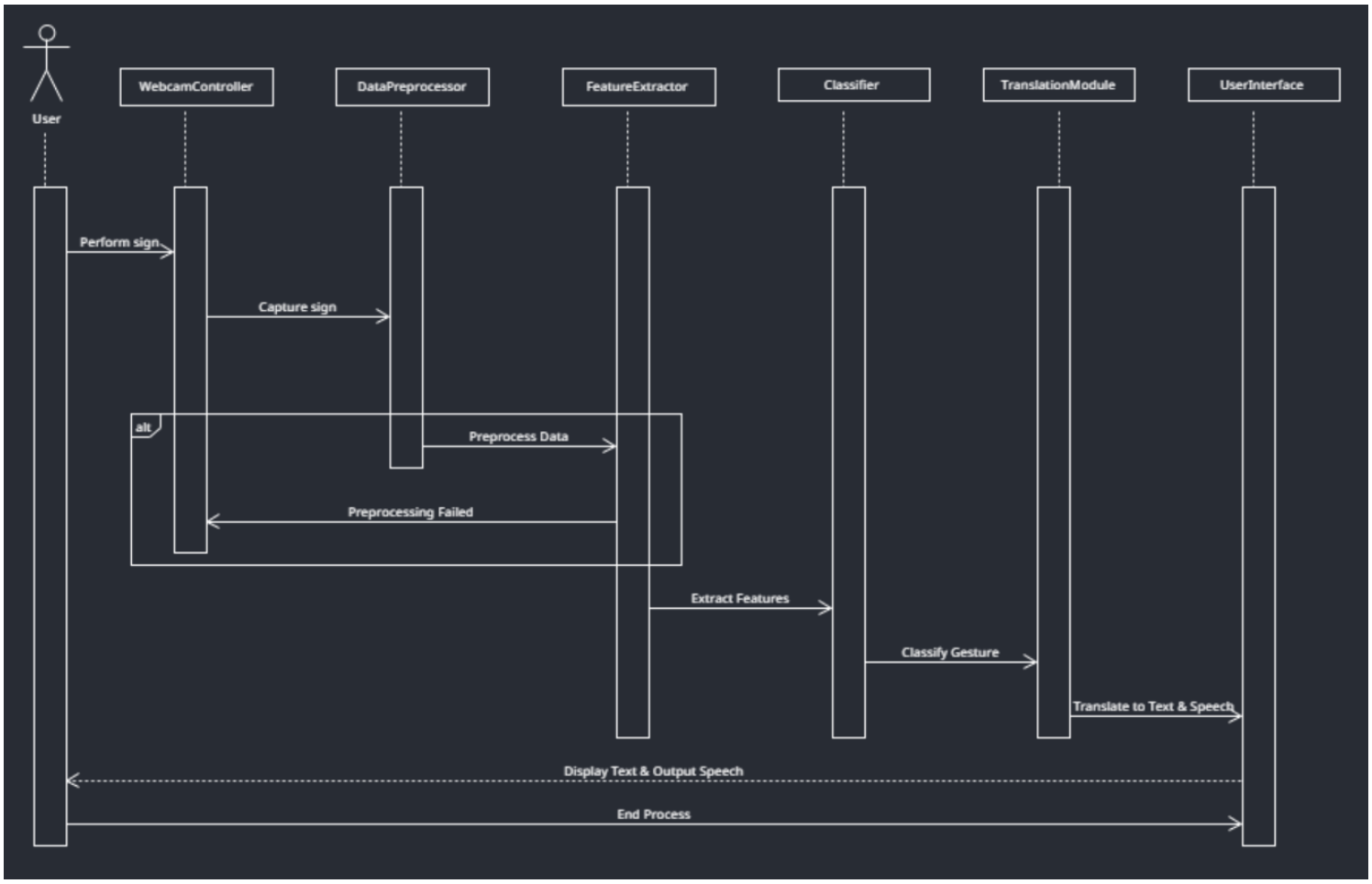


Figure 5: Sequence Diagram

3.5 System Architecture

The project adopts a layered architecture combining computer vision, deep learning and web technologies. The main components include:

Module	Key Technologies	Implementation Details
Data Pipeline	Python (NumPy, OpenCV)	Preprocessing: remove corrupted videos, extract class names, normalize landmarks, augment samples
Hand Tracking	MediaPipe Holistic	Extracts 21 hand + face landmarks with >0.5 confidence

Model Architecture	ConvLSTM2D	Layer 1: ConvLSTM2D (32 filters, 2x2 kernel, tanh activation) with MaxPooling3D and Dropout(0.3) Layer 2: ConvLSTM2D (64 filters) → MaxPooling3D → Dropout Layer 3: ConvLSTM2D (128 filters) → MaxPooling3D → Dropout Layer 4: ConvLSTM2D (256 filters, 1x1 kernel) → Flatten Output: Dense layer with softmax activation for multi-class classification

Table 7: Main Components Table

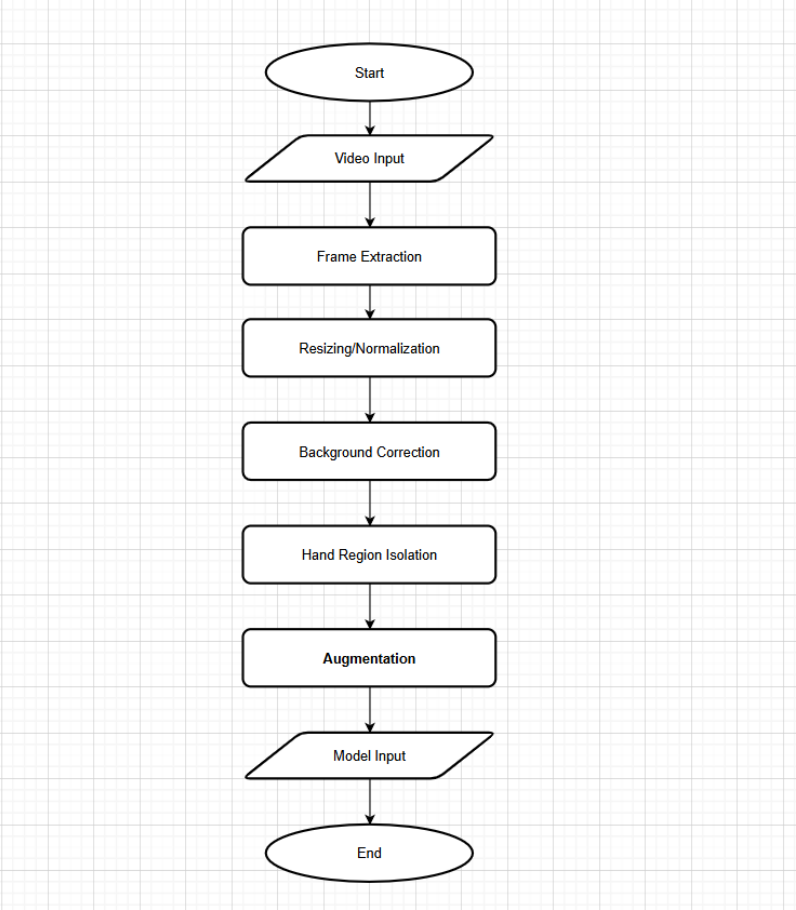


Figure 6: Pre-Processing Flowchart

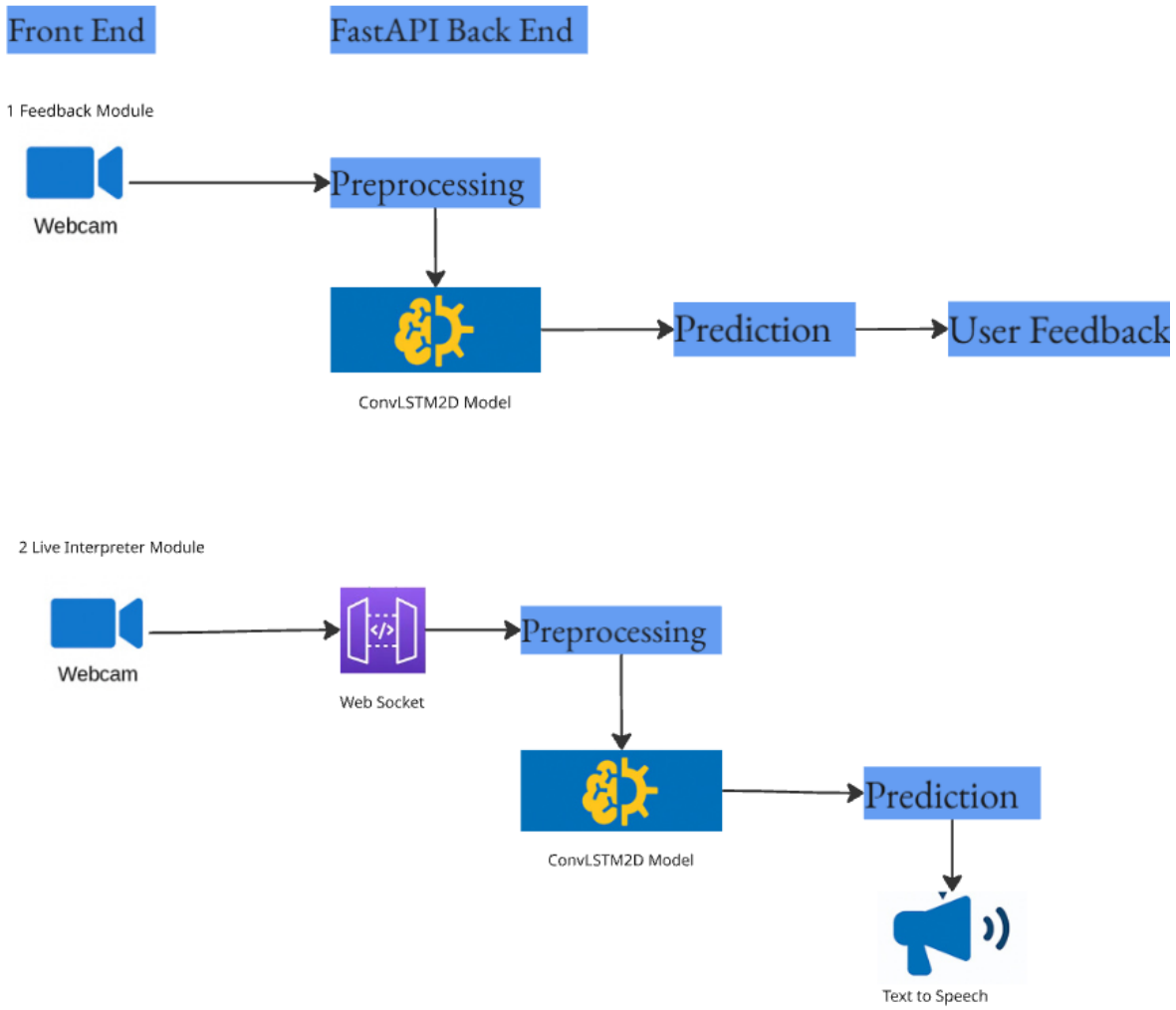


Figure 7: System Architecture Diagram

3.6 Web-based Integration

The web application serves as the user interface for the sign language interpreter. Real-time predictions leverage WebSocket communication for low-latency frame streaming between the frontend and FastAPI backend, ensuring sub-second response times. The subsequent wireframes detail the proposed user interface for the sign language interpreter web application. Developed as simplified visual representations during the early project stages, these blueprints outline the basic layout, navigational structure and the following core functionalities:

- Learning Module: Users browse ASL signs with demo videos.
- Practice Module: Users mimic signs and receive model predictions.
- Model Feedback Module: Users submit corrections to improve model accuracy.
- Live Interpreter Module: WebSocket-driven live gesture recognition.

3.6.1 Wireframes

3.6.1.1 Landing Page

Welcomes users and links to modules.

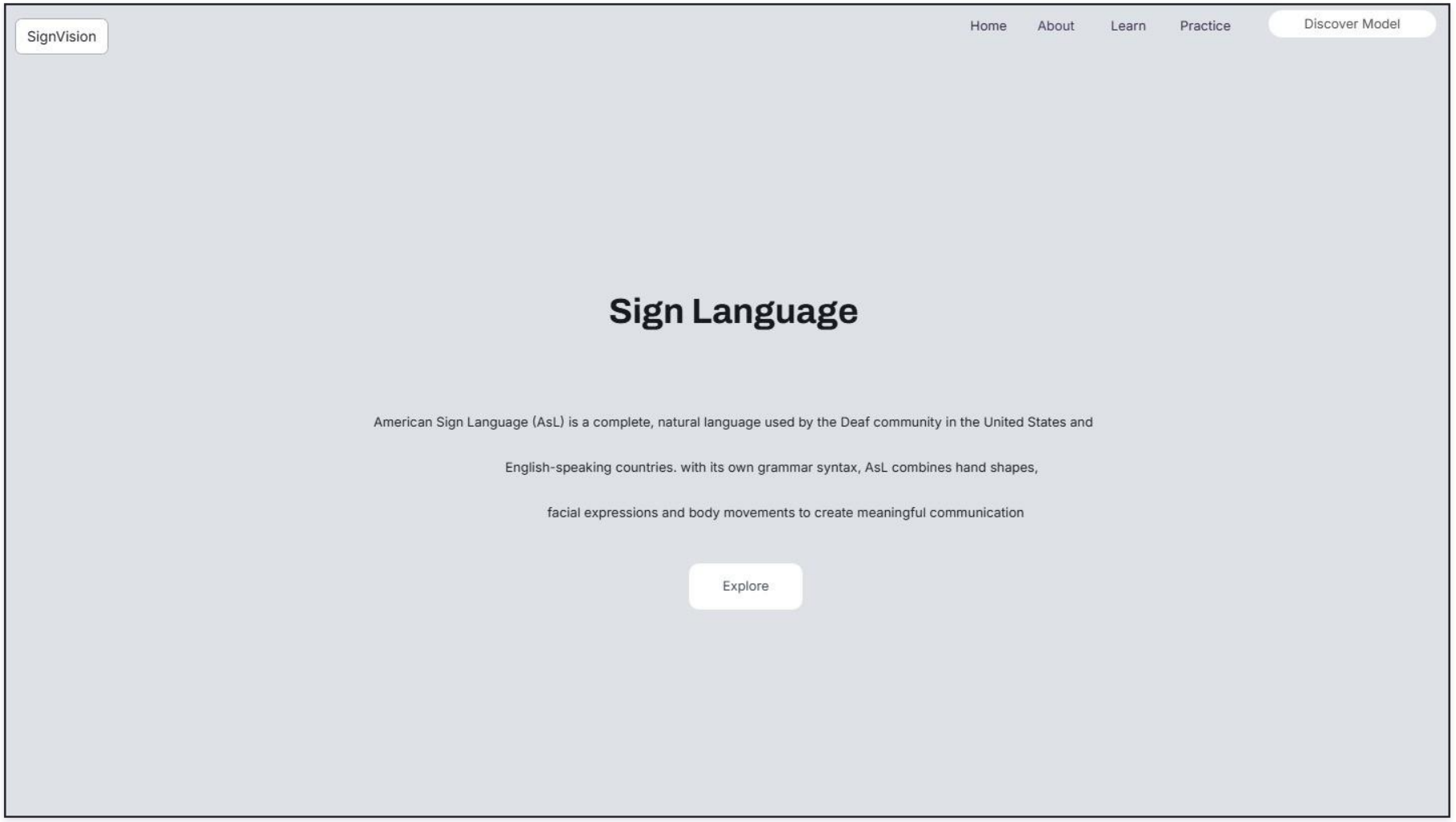


Figure 8: Landing Page

3.6.1.2 About Page

Explains project purpose and background.

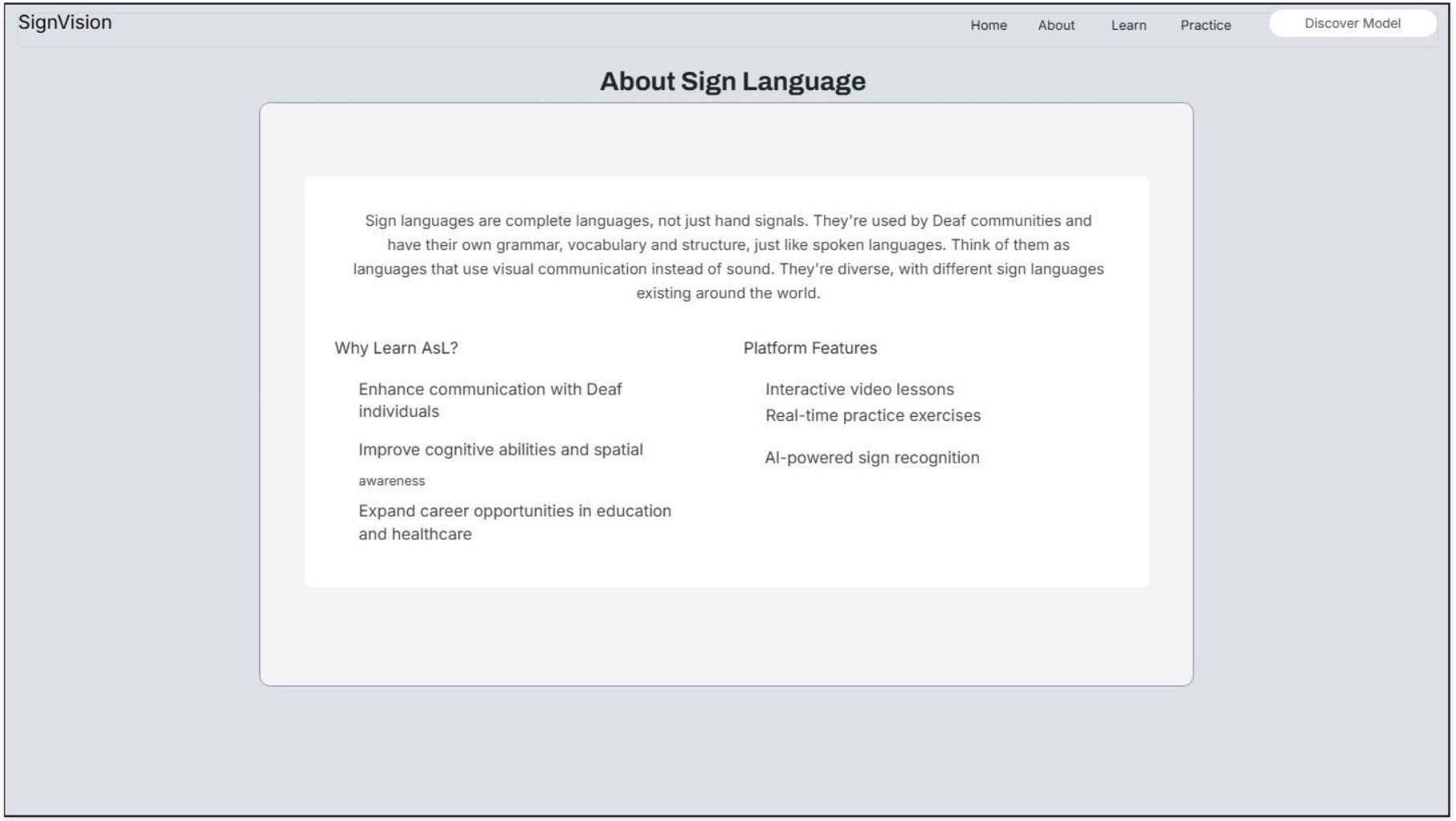


Figure 9: About Page

3.6.1.3 Learn ASL Page

Displays vocabulary list and corresponding sample videos.

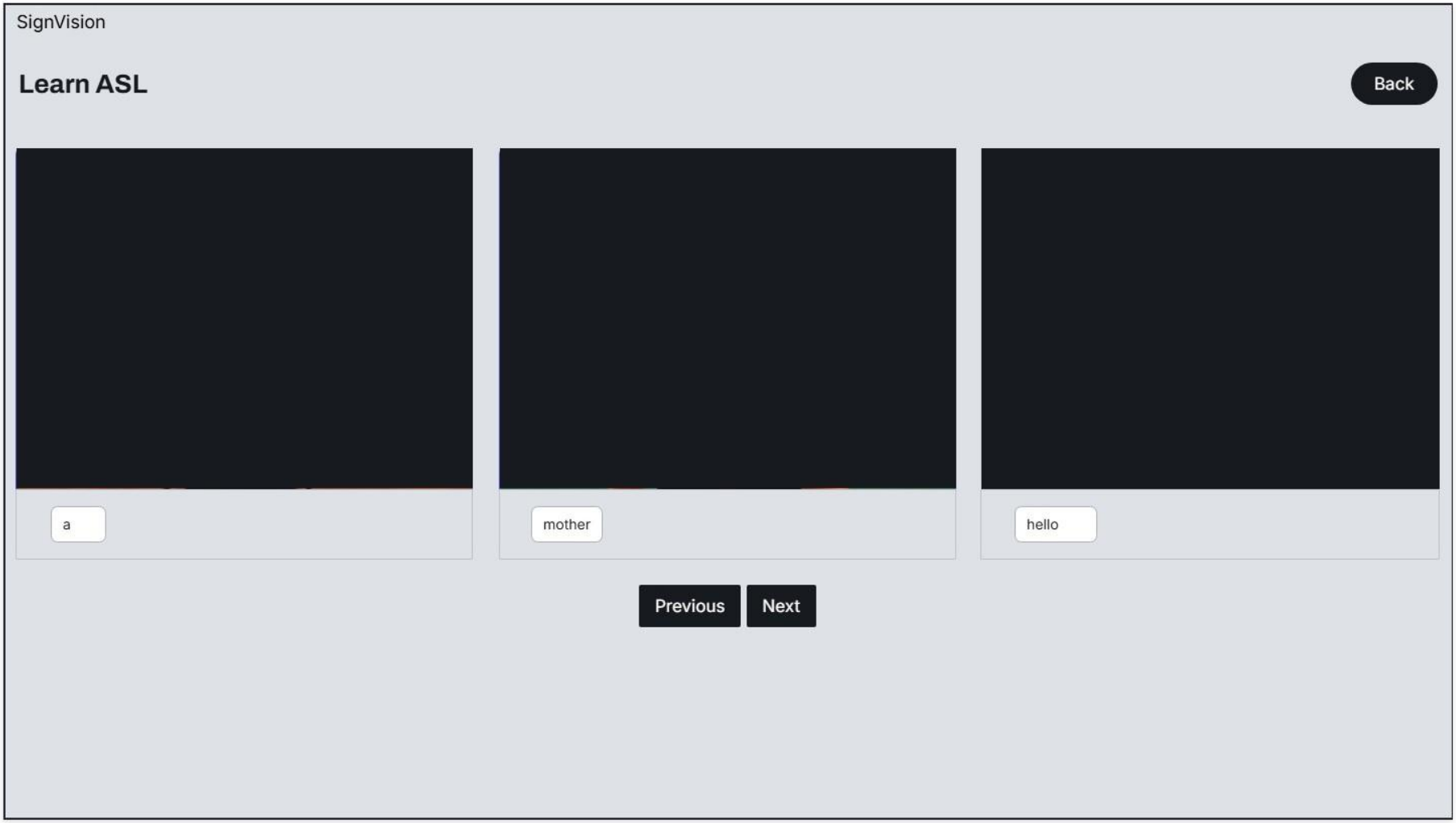


Figure 10: Learn ASL Page

3.6.1.4 Practice ASL Page

Allows user to practice ASL as quiz and displays correct answer.

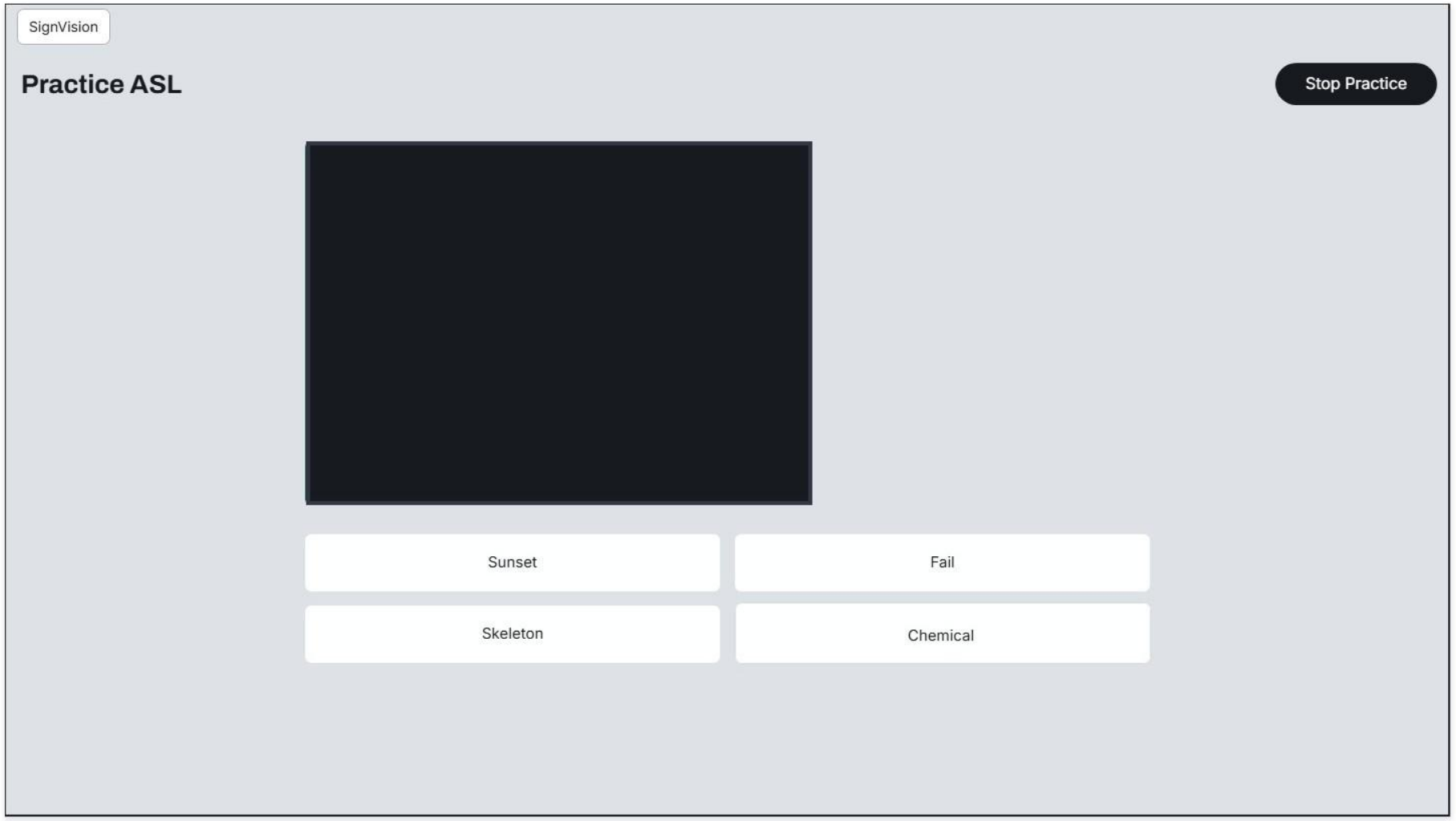


Figure 11: Practice ASL

3.6.1.5 Model Feedback Page

Displays prediction and users mark it correct/incorrect.

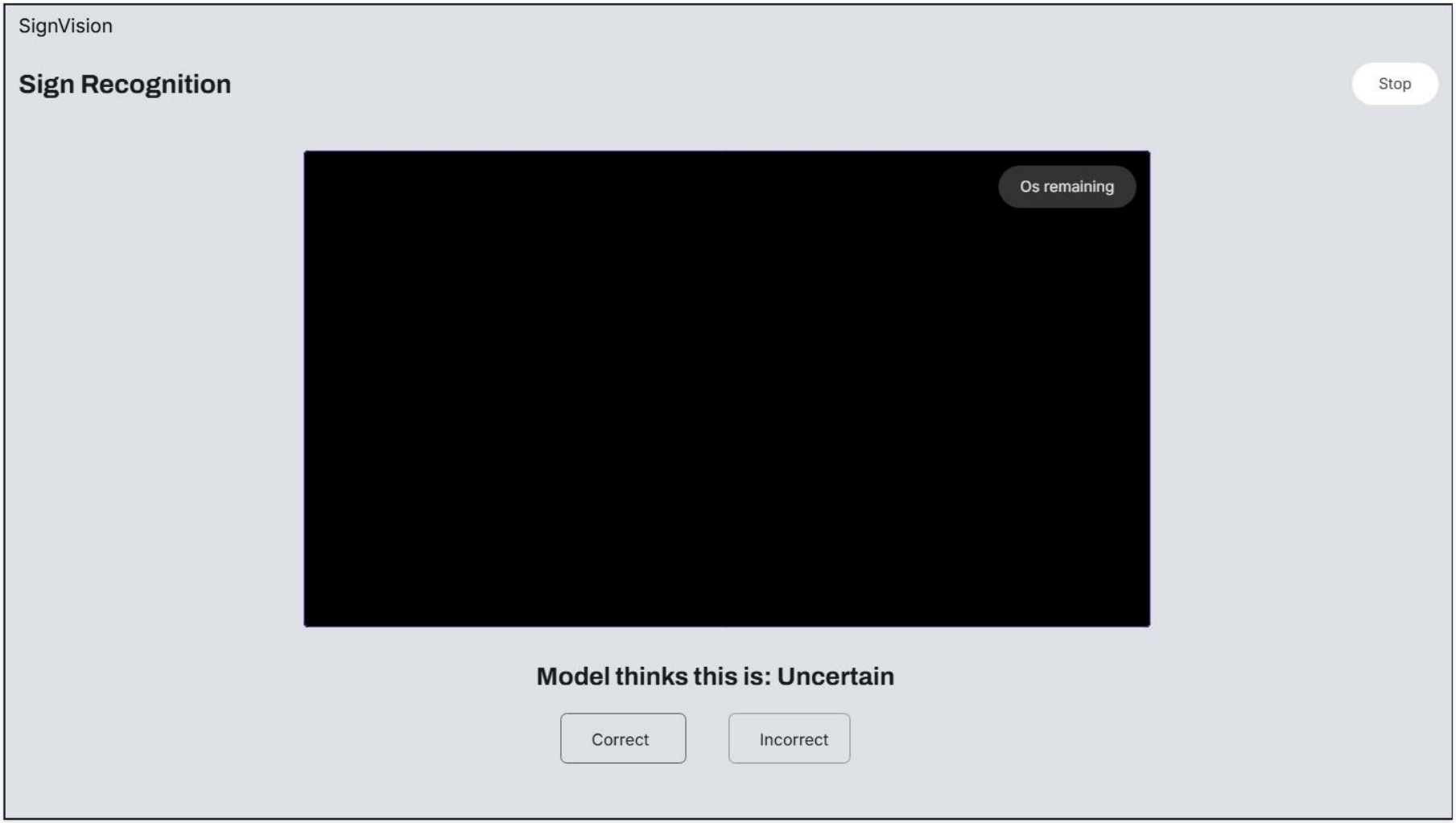


Figure 12: Model Feedback Page

3.6.1.6 Live Interpreter Page

Translates sign gestures to text and speech.

Figure 13: Live Interpreter Page

Chapter 4 – Implementation

This chapter discusses how the model and web application are developed. It covers the complete pipeline from data acquisition and preprocessing to feature extraction, model training and the integration of the prediction system into a web-based application.

4.1 Key Features

- Real-time ASL gesture recognition
- ConvLSTM2D-based deep learning model for spatio-temporal analysis
- MediaPipe Holistic for hand and face landmark extraction
- Web-based frontend (HTML/CSS/JS) with Learn, Practice and Predict modules
- FastAPI backend with REST and WebSocket support
- User feedback collection for future retraining
- Augmented data pipeline for improved generalization

4.2 Project Folder Structure

The system is designed for modularity and clarity, ensuring efficient collaboration and ease of future maintenance. The structure supports the integration of data preprocessing, model development and frontend-backend communication.

The project is organized into the following main directories:

Model/ - Root directory for the model development.

Data Preprocessing/ - Includes all Python scripts responsible for cleaning, organizing and processing raw video data. Scripts here manage tasks like removing corrupted files, normalizing landmarks, extracting class labels and augmenting video samples.

Model Development/ - Contains the core deep learning scripts for training, evaluating, and saving the ConvLSTM2D model. Training logs, loss curves and performance reports are also generated from this folder.

Website/ - Root directory for the web application.

public/ - The main landing page that connects users to the modules.

css/ - Contains styling sheets for page layouts and responsiveness.

js/ - Contains the main JavaScript files that handle client-side interactivity, including WebSocket communications and UI logic.

main.py - The FastAPI entry point that handles backend routes, video preprocessing endpoints, model inference logic and feedback storage.

This structure enables efficient end-to-end functionality, from capturing user video input to predicting and displaying real-time sign interpretations.

4.3 Model Development

4.3.1 Data Acquisition and Data Cleaning

4.3.1.1 Dataset Overview

The primary dataset used for this project is the WLASL-Processed dataset, a structured and annotated collection of ASL video samples, sourced from Kaggle (Baskoro, 2024). It is a refined version of the original WLASL dataset.

The dataset is provided as a JSON file, in which each entry contains:

- The gloss (ASL word label);
- a list of video instances for that gloss; and
- metadata including frames per second (fps), bounding boxes (bbox), frame ranges, signer IDs, video source and the video URL.

Figure 14: Dataset Overview

4.3.1.2 Data Cleaning

The data cleaning process involves several stages to ensure the dataset's integrity and usability. The following steps are:

1. **Downloading videos and structuring folders**

   A Python script is written to automate the download and organization of videos. For each gloss (ASL word), a corresponding directory is created and its corresponding videos are downloaded from the provided URLs. Filenames are sanitized, unsupported formats such as .swf are skipped and failed downloads can be retried up to three times. As a result, the videos are neatly structured into class-specific folders, for further processing and model training.

2. **Filtering video format**

   After downloading, videos are filtered to retain only those with the .mp4 extension, as this format is supported and allows for efficient frame extraction using OpenCV.

3. **Removing corrupted videos**

   Using OpenCV, each video file is opened and the first 10 frames are read. If the file cannot be opened or if the frames are unreadable, the video is flagged as corrupted and removed from the dataset. This ensures that the dataset contains only valid and readable video files, preventing interruptions during training.

4. **Class Reduction**

   The original dataset contains a large number of classes with imbalanced distributions. To improve model performance and reduce noise, classes with very few samples are removed. This results in a reduced dataset with a more balanced class distribution.

5. **Class Label Extraction**

   The class labels are extracted directly from the folder names of the reduced dataset. Each folder represents a specific class, a sign. The class names are cleaned by removing underscores, trimming whitespace and saved the final list to a text file. This list is used as class label reference for training and evaluation.

4.3.2 Data Pre-processing

4.3.2.1 Video Processing and Keypoint Extraction (Feature Extraction and Classification)

After dataset is cleaned and organized, the video processing pipeline reads the ASL video samples from the organized dataset directory where each folder represents a unique sign and contains several videos.

Using MediaPipe's Holistic model, each video is processed frame-by-frame to detect and extract key landmarks for the face and hands. For every frame, 21 landmarks per hand and facial landmarks are retrieved. However, since this implementation focuses on hand gestures, the emphasis is on hand landmarks and face landmarks are used to find nose region and distances are calculated between nose region and hand landmarks and these features are flattened into a fixed-size array to create consistent features. The keypoints are then normalized by computing distances from the nose, used as a reference facial region, to the hand landmarks and the resulting values are flattened into a fixed-size array.

```python
# Function to extract keypoints from the detected hands
def extract_keypoints(results):
    # Extract landmarks
    lh = np.array([[res.x, res.y, res.z] for res in results.left_hand_landmarks.landmark]) if results.left_hand_landmarks else np.zeros((21, 3))
    rh = np.array([[res.x, res.y, res.z] for res in results.right_hand_landmarks.landmark]) if results.right_hand_landmarks else np.zeros((21, 3))
    face = np.array([[res.x, res.y, res.z] for res in results.face_landmarks.landmark]) if results.face_landmarks else np.zeros((468, 3))

    # Face coordinates (use nose as reference point)
    if results.face_landmarks:
        face_point = face[1]
    else:
        face_point = np.zeros(3)

    # Compute distances from face point to each hand landmark
    lh_dist = np.linalg.norm(lh - face_point, axis=1) if lh.any() else np.zeros(21)
    rh_dist = np.linalg.norm(rh - face_point, axis=1) if rh.any() else np.zeros(21)

    # Flatten all features
    return np.concatenate([lh.flatten(), rh.flatten(), lh_dist, rh_dist])
```

Figure 15: Data preprocessing code snippet - Extracting hand and face landmarks

Each video is processed into a sequence of 30 such keypoint frames. This fixed-length sequence provides a uniform input for the ConvLSTM2D model, allowing it to capture both spatial features within each frame and temporal dynamics across the sequence. These processed sequences, along with their corresponding class names, are compiled and saved into .npy files and a CSV file for further analysis. This structured format enables effective feature extraction, serving as the input for the ConvLSTM2D model to classify signs based on spatial and temporal dynamics.

4.3.3 Data Augmentation Techniques

Data augmentation in deep learning is a technique applied to increase the size and diversity of training dataset by modifying data and this helps model to generalize better and prevent overfitting of learning model. Data augmentation techniques are applied during model training by applying random noise and scaling to each video sample.

For each original video sample, ten different variants are generated by perturbing the input data. The added Gaussian noise creates minor changes in the video input such as lighting change and camera shake while the random scaling factor between 0.9 and 1.1 changes distance between user and camera, modifying the hand size. These data augmentation techniques allow to multiply training samples, helps mitigate overfitting and ensures that model learns to focus on the inherent features of the signs rather than artifacts specific to the training set.

```python
# Function to augment the data by applying noise and scaling
def augment_data(X, y):
    augmented_X, augmented_y = [], []
    for i in range(len(X)):
        # Augment each sample by 10 times
        for _ in range(10):
            sample = X[i].copy()

            # Apply random noise
            noise = np.random.normal(0, 0.02, sample.shape)
            augmented_sample = sample + noise

            # Apply slight scaling
            scale_factor = np.random.uniform(0.9, 1.1)
            augmented_sample *= scale_factor

            augmented_X.append(augmented_sample)
            augmented_y.append(y[i])

    return np.array(augmented_X), np.array(augmented_y)
```

Figure 16: Data preprocessing code snippet - Data Augmentation

4.3.4 Model Architecture and Training

The model architecture is based on a deep ConvLSTM2D neural network, designed to learn spatio-temporal patterns from keypoint sequences extracted from the ASL videos. The input to the model is a 5D tensor of shape (samples, 30 frames, 12 rows, 14 columns, 1 channel), which is obtained by reshaping the landmark features extracted during the pre-processing phase. This shape ensures that each video sample is represented as a sequence of 30 frames, where each frame is a 12×14 matrix of features with a single channel.

The model begins with sequential ConvLSTM2D layers that extract both spatial and temporal features. These layers use convolutional units to capture spatial features within each frame, while the LSTM units capture the temporal features across the sequence. This architecture is ideal for gesture recognition, where both the position of hand landmarks and their motion over time are critical.

```
# Function to build and train the ConvLSTM2D model
# Model uses ConvLSTM2D layers for spatio-temporal feature extraction
def train_convlstm2d(X_train, y_train, actions):
    model = Sequential()

    # Add ConvLSTM2D layers with dropout for regularization and max pooling layer for downsampling
    model.add(ConvLSTM2D(filters=32, kernel_size=(2,2), activation='tanh', padding='same',
                         return_sequences=True, input_shape=(30, 12, 14, 1)))
    model.add(MaxPooling3D(pool_size=(1,2,2), padding='same'))
    model.add(TimeDistributed(Dropout(0.3)))

    model.add(ConvLSTM2D(filters=64, kernel_size=(2,2), activation='tanh', padding='same', return_sequences=True))
    model.add(MaxPooling3D(pool_size=(1,2,2), padding='same'))
    model.add(TimeDistributed(Dropout(0.3)))

    model.add(ConvLSTM2D(filters=128, kernel_size=(2,2), activation='tanh', padding='same', return_sequences=True))
    model.add(MaxPooling3D(pool_size=(1,2,2), padding='same'))
    model.add(TimeDistributed(Dropout(0.3)))

    model.add(ConvLSTM2D(filters=256, kernel_size=(1,1), activation='tanh', return_sequences=False))
    model.add(Flatten())
    model.add(Dense(len(actions), activation='softmax'))

    model.compile(optimizer='Adam', loss='sparse_categorical_crossentropy', metrics=['accuracy'])

    early_stopping = EarlyStopping(monitor='val_loss', patience=10, restore_best_weights=True)

    history = model.fit(X_train, y_train, epochs=100, validation_split=0.2, callbacks=[early_stopping])

    return model, history
```

Figure 17: Model code snippet

The first ConvLSTM2D layer contains 32 filters and a 2×2 kernel. It uses the tanh activation function for smoother gradient flow in sequential data. It is followed by a MaxPooling3D layer to reduce spatial dimensions and prevent overfitting. Dropout is applied in a TimeDistributed manner after each convolutional block to prevent overfitting by randomly deactivating a

fraction of the neurons during training. This pattern continues with ConvLSTM2D layers of increasing filter sizes: 64, 128 and 256.

The final ConvLSTM2D block uses a 1×1 kernel to compress the learned features before flattening. The output layer is a Dense softmax layer, responsible for producing class probabilities across all sign classes.

The final layers flatten the output from the convolutional blocks and pass the features through a dense layer with softmax activation, producing classification probabilities for each of the ASL sign classes.

The model is compiled using the Adam optimizer for its adaptive learning rate and fast convergence. Sparse categorical cross-entropy is used as the loss function, making it ideal for multi-class problems where the output labels are integer-encoded. To prevent overfitting and reduce training time, EarlyStopping is applied with a patience of 10 epochs during training by monitoring the validation loss and restore the best model weights if no improvement is observed for the predefined number of epochs. The model's performance is evaluated using accuracy and validation loss and the best weights are restored automatically.

The training process spans up to 100 epochs, using an 80/20 split between training and validation data. Accuracy and loss trends are plotted using Matplotlib and model predictions, classification reports and confusion matrices are saved for analysis.

The final output includes a saved model file (trained_model_reduced_dataset.h5), visualizations (training_accuracy.png, training_loss.png), classification reports (classification_report.csv, prediction_details.csv) and confusion matrices split into two visuals for better interpretability.

4.4 Web Application Development

4.4.1 Website Development

A responsive UI is developed using HTML, CSS and JavaScript. Pages include Learn, Practice, Feedback modules and real-time interpreter. Real implementation screenshots are included in the Appendix.

4.4.2 Learning Functionality

The learning module is designed to provide instructional videos for sign language. Videos are organized by sign, allowing users to learn the correct form and movement associated with each sign. Users can pause, replay and navigate through the content at their own pace. These features are integrated into the website design to encourage continuous learning and improve overall comprehension of sign language.

4.4.3 Practice Functionality

The practice module provides users with an interactive environment to test their knowledge. Users are presented with a video of a sign and must select the correct meaning from multiple choices. The system provides immediate feedback on their responses, highlighting both correct and incorrect choices. The practice module is designed to be engaging and informative.

4.4.4 Integration of Model with Front-End and Feedback Collection

The integration of the ConvLSTM2D model with the front-end is handled by a FastAPI server that uses several endpoints. When a user uploads a video for sign prediction, the video is processed through the pre-processing pipeline and the model generates a prediction. The prediction, along with a confidence score, is returned to the front-end and this result is then dynamically rendered on the user interface. Users have the option to provide feedback if the prediction is inaccurate. The feedback endpoint captures the video along with the correct sign and predicted sign. User feedback (correct/incorrect predictions) is stored for periodic offline retraining, enabling iterative model improvement without real-time adaptation.

4.4.5 Real-Time Predictions

The frontend and backend are connected via WebSocket. Real-time prediction is achieved by continuously capturing video frames from the user's webcam. Every 3-second window (30 frames) is analyzed and sent to the backend for classification. The prediction result is then rendered on the user interface for 2 seconds before the next cycle starts and thus allowing for continuous gesture interpretation.

4.5 System Summary

System	AI-Based Sign Language Recognition (Web-based)
Key Features & Differences	Real-time gesture recognition, spatio-temporal analysis using ConvLSTM2D, runs in-browser with FastAPI backend
Hardware Requirement	Standard webcam, Mid-range CPU
Accessibility	High, Browser-based
Accuracy	95
Training Method Used	ConvLSTM2D
Model Training Time (hrs)	20
Testing Method Used	Webcam-based gesture testing in simulated use case
Testing Efficiency	Moderate
Latency (ms)	~500ms (including video decoding, landmark extraction, inference)
UI Development	Responsive UI using HTML, CSS, JavaScript
Functional Modules	Learn, Practice, Feedback, Real-Time Interpreter

Table 8: System Summary

4.6 Challenges Encountered

As the project progresses, several challenges arise that impact the development and implementation of the AI-driven sign language interpretation system. These challenges include:

- Environmental Externalities: Overall changes in lightning, cluttered backgrounds and alteration in camera positions can make it difficult to accurately recognize gestures. Moreover, their differences under the environment settings add more issues. Background Identification, Lighting Change Normalization and Region Isolation are pre-processing steps that are processed to overcome these obstacles.

- Immediate Reaction Capability: Deep learning models can put an excessive load on computational resources, which slower systems struggle with, especially if dealing with everyday devices. The need to maintain accuracy when optimizing additional computation force the use of model quantization, more efficient designs and the addition of GPUs or TPUs for processing power.

- User Variability: The patterns of hand movements are unique to every individual when performing sign language. This diversity has the potential to impact the model's adaptiveness across different users, often resulting in inaccurate interpretations. Besides, other aspects like hand occlusions (where one hand covers the other hand) and different signers' angles of body position need to be considered.

- Dataset Limitations: There is not enough data to support the claim of high-quality datasets being diverse. Most of the available datasets have inadequate variety in signer's ethnicity, amount of light per signing scene and signing continuity.

- Integration with Web-Based Systems: Adopting the AI sign language interpreter inside a website brought a whole new level of technical issues like increased latency, server overload and things like processing videos in real time. Instead of HTTP, video are streamed through WebSockets.

- Ethical and Cultural Considerations: Ensuring that the system respects the cultural nuances of sign language and the preferences of the deaf community is challenging. The AI-based interpreter is designed to complement human interpreters rather than replace them.

Chapter 5 – Testing

5.1 Testing Procedures

To validate the robustness, accuracy and usability of the system, multiple testing methodologies are employed. Functional testing is conducted to confirm that key features such as real-time prediction, learning modules and practice quizzes behave according to specifications. Each user flow is manually examined, including gesture recording, prediction response display and feedback collection.

Unit testing is applied to individual components such as MediaPipe keypoint extraction, data augmentation functions and model prediction functions. This ensures that even when isolated, these units perform correctly and return expected outputs for various test inputs.

Test Case	Description	Outcome
Data Preprocessing	Verify fixed 30-frame sequence length across inputs	Consistent 30-frame sequences per gesture
MediaPipe Keypoint Extraction	Test if landmarks are extracted in varied lighting conditions	21 hand landmarks consistently extracted
Model Prediction	Check if valid class label is returned from ConvLSTM2D model	Predicted class label within predefined class set
Model Inference Latency	Measure prediction time per 30-frame input	Inference time ≤ 2s for smooth real-time output
Feedback Submission	Validate video is stored under the correct folder based on prediction	Video saved in folder named after the correct predicted class label
WebSocket Stability	Assess real-time stream performance under network latency	Minimal frame drop, consistent prediction loop cycle

Table 9: Unit Testing

Performance testing assesses the system's response time and computational efficiency. The FastAPI backend is evaluated under load conditions to ensure that the system could maintain real-time responsiveness. The ConvLSTM2D model is evaluated to ensure it could produce predictions in under two seconds per sequence.

Integration testing verifies end-to-end functionality between the frontend, backend and model prediction layers. It ensures data captured by the webcam is processed correctly, transmitted via WebSocket, interpreted by the ConvLSTM2D model, and that predictions are returned and displayed without loss of fidelity or timing delays.

Edge case testing is conducted by introducing invalid gestures, partial hand visibility and no movement input. The system was observed to ignore or classify these as 'Unknown' with no crashes or unexpected behavior.

Additionally, model testing is aligned with evaluation metrics detailed in Chapter 6, including classification accuracy, confusion matrix distribution and real-time prediction quality.

5.2 Frameworks Applied

The system is evaluated using both standard machine learning evaluation frameworks and software testing paradigms. For model assessment, classification reports and confusion matrices are generated using scikit-learn. These reports provide precision, recall, F1-score and accuracy metrics per class, enabling fine-grained performance analysis.

Matplotlib and Seaborn are used to visualize training history and confusion matrices. Real-time testing tools like browser developer consoles and FastAPI logs are used to monitor response times and detect potential latency or error spikes.

TensorFlow/Keras are used to build and train the ConvLSTM2D deep learning architecture. Keras callbacks such as EarlyStopping and ModelCheckpoint are used during training to prevent overfitting and ensure best model retention.

OpenCV is employed for low-level video processing during unit tests and integration testing. It is used to validate frame integrity, frame rates and detect corrupted or empty video streams during preprocessing and feedback submission.

MediaPipe is primarily used for landmark extraction and MediaPipe’s built-in confidence score filtering helps ensure that only high-quality frames (confidence > 0.5) are used during preprocessing and real-time inference, which indirectly supports the model's reliability during testing phases.

FastAPI’s Built-in TestClient is used for backend API testing. This enables simulated POST requests to endpoints (for example, prediction and feedback submission), allowing validation of API response status codes, content formats and latency under test conditions.

GitHub is used as the version control platform throughout development. It enables systematic tracking of model changes, experiment branches and testing iterations. GitHub Pull Requests document versioned improvements and testing status before merging.

Chapter 6 – Evaluation

This chapter presents an evaluation of the AI-based sign language recognition model, with a focus on its performance metrics, the reasoning behind the chosen methodology and the broader implications of the results. The aim is to assess the model's effectiveness in interpreting sign gestures accurately across 120 classes and to understand the challenges posed by real-world scenarios, including class imbalance and gesture similarity.

6.1 Purpose of the Chosen Methodology

The methodology adopted in this research integrates temporal and spatial features using a hybrid ConvLSTM2D architecture. This approach is preferred due to its proven capability to handle spatiotemporal data effectively in gesture recognition tasks (Shi et al., 2015). Convolutional layers capture spatial dependencies within frames, while LSTM layers model the temporal dynamics of gestures across frames, making it well-suited for recognizing sequences of hand movements in sign language. The primary objective was to develop a robust, real-time system capable of interpreting sign gestures and converting them into textual and audible feedback.

The selection of the ConvLSTM2D architecture over conventional CNNs or 3D CNNs was due to the nature of sign language, which incorporates both spatial and temporal dynamics. While standard CNNs are proficient in extracting spatial features from static images, they lack the capacity to model sequential dependencies critical for gesture transitions. On the other hand, 3D CNNs, although capable of learning spatio-temporal features, are computationally intensive and less efficient for real-time applications due to their heavy memory requirements and slow inference speeds. ConvLSTM2D presents an optimal trade-off by embedding convolutional operations within LSTM cells, allowing it to effectively capture temporal evolution in spatial patterns across video frames. This results in better performance for continuous gesture recognition while maintaining acceptable inference latency suitable for real-time web-based deployment.

MediaPipe Holistic is selected over other pose estimation frameworks such as OpenPose due to its high efficiency, ease of integration and reduced computational overhead. MediaPipe operates with a unified pipeline for hand, face and body landmark detection, offering real-time tracking capabilities even on CPU-based systems. Compared to OpenPose, which is known for high accuracy but often requires GPU support for smooth

operation, MediaPipe achieves comparable results at a fraction of the resource cost. Its native support for Python and integration with OpenCV further simplified deployment within the backend system, contributing to lower latency and better responsiveness of the sign language interpreter prototype.

A key consideration in this project is the trade-off between model accuracy and latency in real-time applications. High-accuracy models are often deeper and more complex, which significantly increases processing time and hinders responsiveness. To mitigate this, the model is designed with lightweight ConvLSTM2D blocks and inference performance is further optimized through input resolution reduction and selective frame sampling. While this results in a marginal decrease in overall accuracy compared to more complex architectures such as Transformers, it enables inference with imperceptible delay. This trade-off prioritizes user experience and system accessibility, making the model more suitable for real-time web-based sign language interpretation.

6.2 Evaluation Method

The evaluation follows a structured pipeline comprising the following steps:

1. Dataset Splitting: The dataset is split into training (85%), validation (10%) and test (5%) subsets to ensure unbiased model assessment on unseen data.

2. Keypoint Extraction: MediaPipe is used to extract hand and face landmarks from video sequences. Each 3-second gesture sequence (30 frames) are processed to generate a feature matrix.

3. Training and Validation: The model is trained over 100 epochs with early stopping based on validation loss. Accuracy and loss metrics are monitored throughout.

4. Prediction and Metrics: After training, predictions on the test set are stored in CSV format. Evaluation metrics including accuracy, precision, recall and F1-score are generated using classification_report from scikit-learn.

5. Visualization: Performance is further analyzed using plotted training and validation curves and confusion matrices to identify class-level insights.

6. Ethical Considerations: A consent modal ensures users explicitly agree to data collection before using the feedback feature.

6.3 Model Performance Evaluation

The model is evaluated using a reserved 5% test subset, ensuring that the performance metrics reflect its ability to generalize to unseen data. Predictions are saved to a CSV file for further analysis. The evaluation of the model's performance is based on metrics such as accuracy, precision and recall. Accuracy measures the overall correctness, while precision indicates how many of the predicted positive cases are actually correct. These metrics are essential for identifying both the strengths and limitations of the model, particularly in the context of real-time gesture recognition where misclassification of signs can impact communication quality.

6.3.1 Accuracy, Precision Analysis

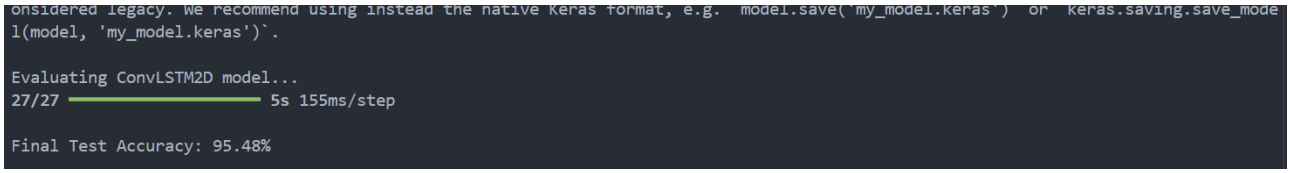


Figure 18: Model Accuracy Percentage

The model achieved a validation accuracy of 95.48%, indicating strong generalisation across 120 gesture classes.

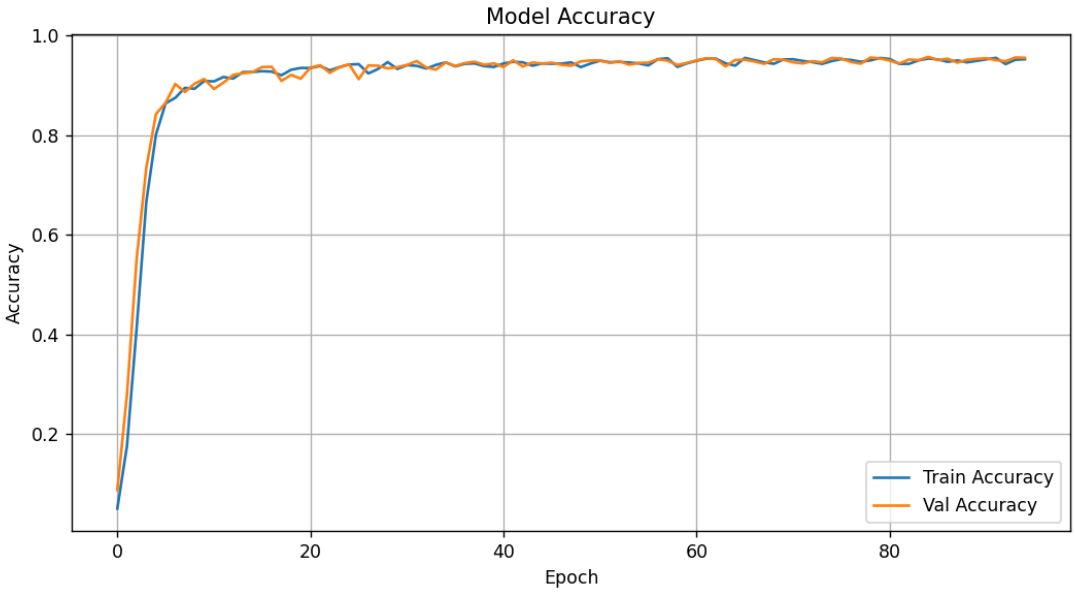


Figure 19: Model Accuracy graph

The training accuracy and validation accuracy curves (Figure 20) show a consistent upward trend, both reaching values close to 0.99 by the 90th epoch. The model rapidly achieves over 90% accuracy within the first 10 epochs, demonstrating effective learning from the start. The convergence of training and validation curves suggests a minimal overfitting, indicating that the model generalises well across both seen and unseen data.

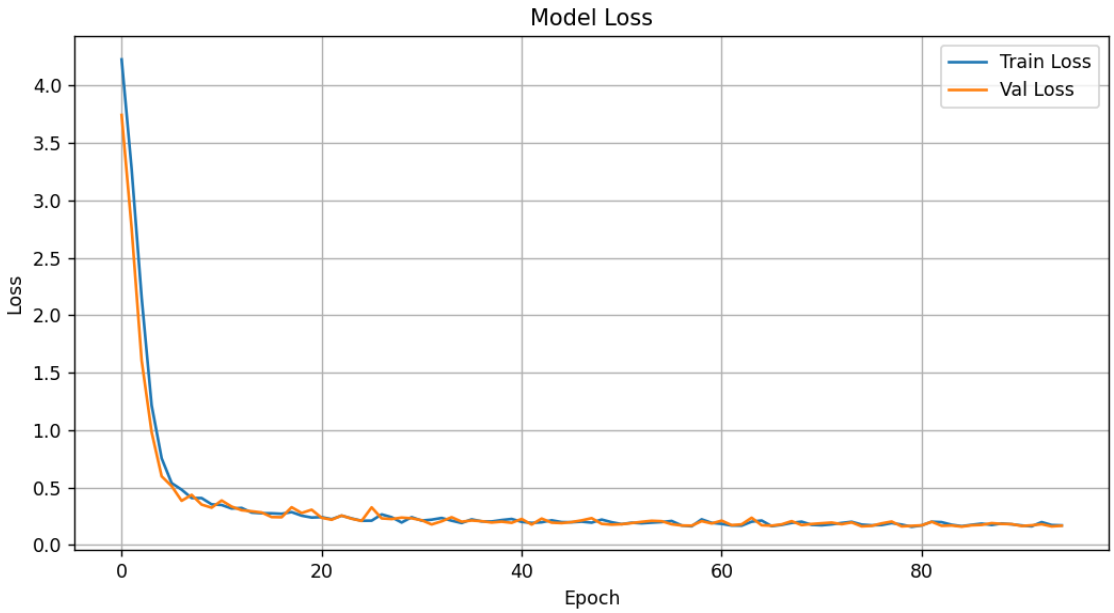


Figure 20: Model Loss Graph

The loss curves (Figure 21) for both training and validation data show a rapid decline within the initial epochs, with the loss dropping from over 4.0 to below 0.2. Beyond epoch 20, the loss stabilises and both curves remain closely aligned, further confirming the model's ability to avoid overfitting.

The evaluation compares the model's predictions with the actual labels and uses scikit-learn's classification_report to generate metrics. These metrics help to pinpoint specific classes where the model performs poorly and hence guiding further refinements. Visualizations are created to illustrate these metrics.

Most classes, such as "apple", "argue", "appointment" and "basketball", achieved a perfect score of 1.00 across precision, recall and F1-score, highlighting the model's strong ability to differentiate these signs.

A few classes, such as "before" and "candy", show lower performance. For instance, "before" has precision, recall and F1-score of 0.67, due to a limited number of samples (support = 3) or high similarity with other gestures.

Despite some variability, the overall performance remains strong, with the majority of classes exceeding 0.90 in all metrics.

This detailed analysis enables the identification of potentially ambiguous or underrepresented signs and supports future improvements such as class balancing or refined feature extraction.

```
$ python classification_report.py
              precision    recall  f1-score   support

      accident       0.92      1.00      0.96        12
           add       1.00      0.86      0.92         7
        africa       1.00      0.89      0.94         9
           ago       0.83      0.71      0.77         7
         alone       1.00      1.00      1.00         7
         apple       1.00      1.00      1.00         9
   appointment       1.00      1.00      1.00         8
         argue       1.00      1.00      1.00        11
      australia       1.00      1.00      1.00         3
           bad       1.00      1.00      1.00         8
       balance       1.00      1.00      1.00         4
           bar       1.00      1.00      1.00         8
        barely       1.00      1.00      1.00         8
    basketball       1.00      1.00      1.00         8
         beard       1.00      1.00      1.00         9
           bed       1.00      1.00      1.00        12
        before       0.67      0.67      0.67         3
         black       1.00      1.00      1.00         3
       bowling       1.00      1.00      1.00         8
       brother       1.00      1.00      1.00         5
    california       1.00      1.00      1.00         4
          call       1.00      1.00      1.00         7
         candy       1.00      0.64      0.78        11
       careful       1.00      1.00      1.00         4
        carrot       0.92      1.00      0.96        11
      champion       1.00      1.00      1.00         5
        change       1.00      1.00      1.00         9
```

Figure 21: Classification report (1)

```
      champion       1.00      1.00      1.00         5
        change       1.00      1.00      1.00         9
          chat       1.00      1.00      1.00         4
         cheat       1.00      1.00      1.00         8
         check       1.00      1.00      1.00        10
          cold       1.00      1.00      1.00         6
      computer       1.00      1.00      1.00         8
      convince       1.00      1.00      1.00         6
          cool       1.00      1.00      1.00         9
          corn       1.00      0.92      0.96        12
        cousin       1.00      0.93      0.97        15
           cry       1.00      1.00      1.00         3
          dark       1.00      0.91      0.95        11
      daughter       1.00      1.00      1.00        12
          deaf       1.00      1.00      1.00         3
      decorate       1.00      1.00      1.00         4
         delay       0.83      1.00      0.91         5
     delicious       1.00      1.00      1.00         8
          dive       1.00      1.00      1.00         8
           dog       1.00      1.00      1.00         1
         drink       1.00      1.00      1.00         7
        family       1.00      1.00      1.00         9
           far       1.00      1.00      1.00         9
          fast       1.00      1.00      1.00         4
           fat       1.00      1.00      1.00         6
        finish       1.00      1.00      1.00         3
         first       1.00      1.00      1.00         6
        follow       1.00      1.00      1.00         5
          full       1.00      1.00      1.00         5
        future       1.00      1.00      1.00         4
          give       1.00      1.00      1.00         5
```

Figure 22: Classification report (2)

future	1.00	1.00	1.00	4
give	1.00	1.00	1.00	5
glasses	1.00	1.00	1.00	7
go	1.00	1.00	1.00	11
good	1.00	1.00	1.00	1
graduate	1.00	1.00	1.00	9
great	1.00	1.00	1.00	8
hearing	0.89	0.89	0.89	9
help	1.00	1.00	1.00	10
hope	1.00	1.00	1.00	12
hot	1.00	1.00	1.00	3
inform	1.00	1.00	1.00	10
language	1.00	1.00	1.00	4
last	1.00	0.80	0.89	5
later	1.00	0.80	0.89	5
laugh	1.00	1.00	1.00	10
leave	1.00	1.00	1.00	5
letter	1.00	1.00	1.00	7
make	1.00	1.00	1.00	7
man	0.36	1.00	0.53	14
many	1.00	1.00	1.00	2
meet	1.00	0.86	0.92	7
mother	1.00	1.00	1.00	9
move	1.00	1.00	1.00	5
necklace	1.00	1.00	1.00	7
new	1.00	1.00	1.00	2
no	1.00	1.00	1.00	2
now	1.00	0.92	0.96	12
pizza	1.00	1.00	1.00	11
play	1.00	1.00	1.00	4
postpone	1.00	0.83	0.91	6

Figure 23: Classification report (3)

postpone	1.00	0.83	0.91	6
pull	1.00	0.89	0.94	9
ready	1.00	0.75	0.86	4
room	0.89	1.00	0.94	8
same	1.00	1.00	1.00	3
sandwich	1.00	1.00	1.00	7
score	1.00	1.00	1.00	9
secretary	1.00	1.00	1.00	4
shirt	1.00	0.91	0.95	11
short	1.00	0.92	0.96	12
silly	1.00	1.00	1.00	10
son	1.00	1.00	1.00	3
stay	1.00	0.83	0.91	6
take	1.00	1.00	1.00	2
tall	1.00	0.86	0.92	7
tell	0.80	0.67	0.73	6
thanksgiving	1.00	1.00	1.00	8
thin	0.91	0.77	0.83	13
thursday	1.00	1.00	1.00	4
toast	1.00	1.00	1.00	9
trade	1.00	1.00	1.00	9
upset	1.00	1.00	1.00	10
wait	1.00	1.00	1.00	9
walk	1.00	1.00	1.00	6
wet	1.00	0.86	0.92	7
what	1.00	1.00	1.00	5
white	1.00	0.75	0.86	4
who	0.73	0.62	0.67	13
why	0.50	0.50	0.50	2
woman	1.00	0.80	0.89	5
work	1.00	1.00	1.00	8

Figure 24: Classification report (4)

stay	1.00	0.83	0.91	6
take	1.00	1.00	1.00	2
tall	1.00	0.86	0.92	7
tell	0.80	0.67	0.73	6
thanksgiving	1.00	1.00	1.00	8
thin	0.91	0.77	0.83	13
thursday	1.00	1.00	1.00	4
toast	1.00	1.00	1.00	9
trade	1.00	1.00	1.00	9
upset	1.00	1.00	1.00	10
wait	1.00	1.00	1.00	9
walk	1.00	1.00	1.00	6
wet	1.00	0.86	0.92	7
what	1.00	1.00	1.00	5
white	1.00	0.75	0.86	4
who	0.73	0.62	0.67	13
why	0.50	0.50	0.50	2
woman	1.00	0.80	0.89	5
work	1.00	1.00	1.00	8
wow	1.00	1.00	1.00	8
write	1.00	1.00	1.00	4
year	1.00	1.00	1.00	6
yes	1.00	1.00	1.00	5
yesterday	1.00	1.00	1.00	4
accuracy			0.95	840
macro avg	0.98	0.96	0.96	840
weighted avg	0.97	0.95	0.96	840

```
Classification report saved as 'classification_report.csv'
```

Figure 25: Classification report (5)

6.3.2 Confusion Matrix Analysis

A confusion matrix is used to visualize the model's classification performance by comparing predicted labels against actual labels. Each cell in the matrix shows the number of samples that fall into the corresponding true versus predicted category, the number of samples predicted as a certain class (columns) versus their true labels (rows). This matrix helps identify which signs are often confused with one another and provides insights into potential issues with specific class boundaries.

For example, if similar signs are frequently misclassified, this could indicate the need for more robust feature extraction or additional data augmentation for those classes. The confusion matrix is generated using Matplotlib and Seaborn.

The Confusion matrices (Figures 27 and 28) below are used to further explore model predictions and understand class-wise misclassifications.

For well-separated signs like "bed", "call" and "carrot", the confusion matrices show a clear diagonal, indicating excellent classification accuracy.

Misclassifications are more likely in signs with lower support (fewer samples) or signs that share similar gestures. For instance, gestures like "add" and "ago" may have visual similarities that confuse the model, as suggested by their lower F1-scores.

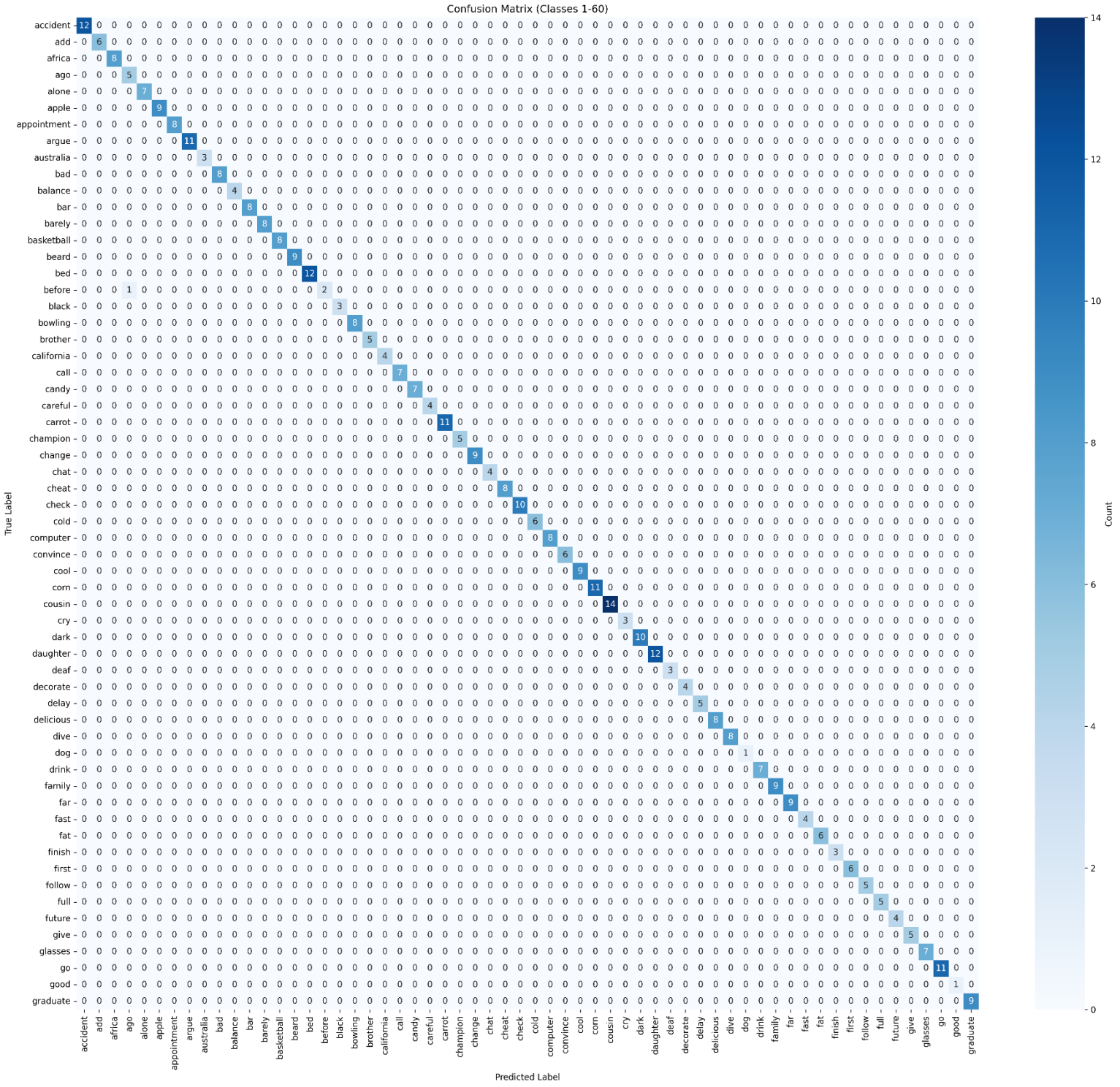


Figure 26: [Confusion Matrix 1-60] (confusion_matrix_1-60.png)

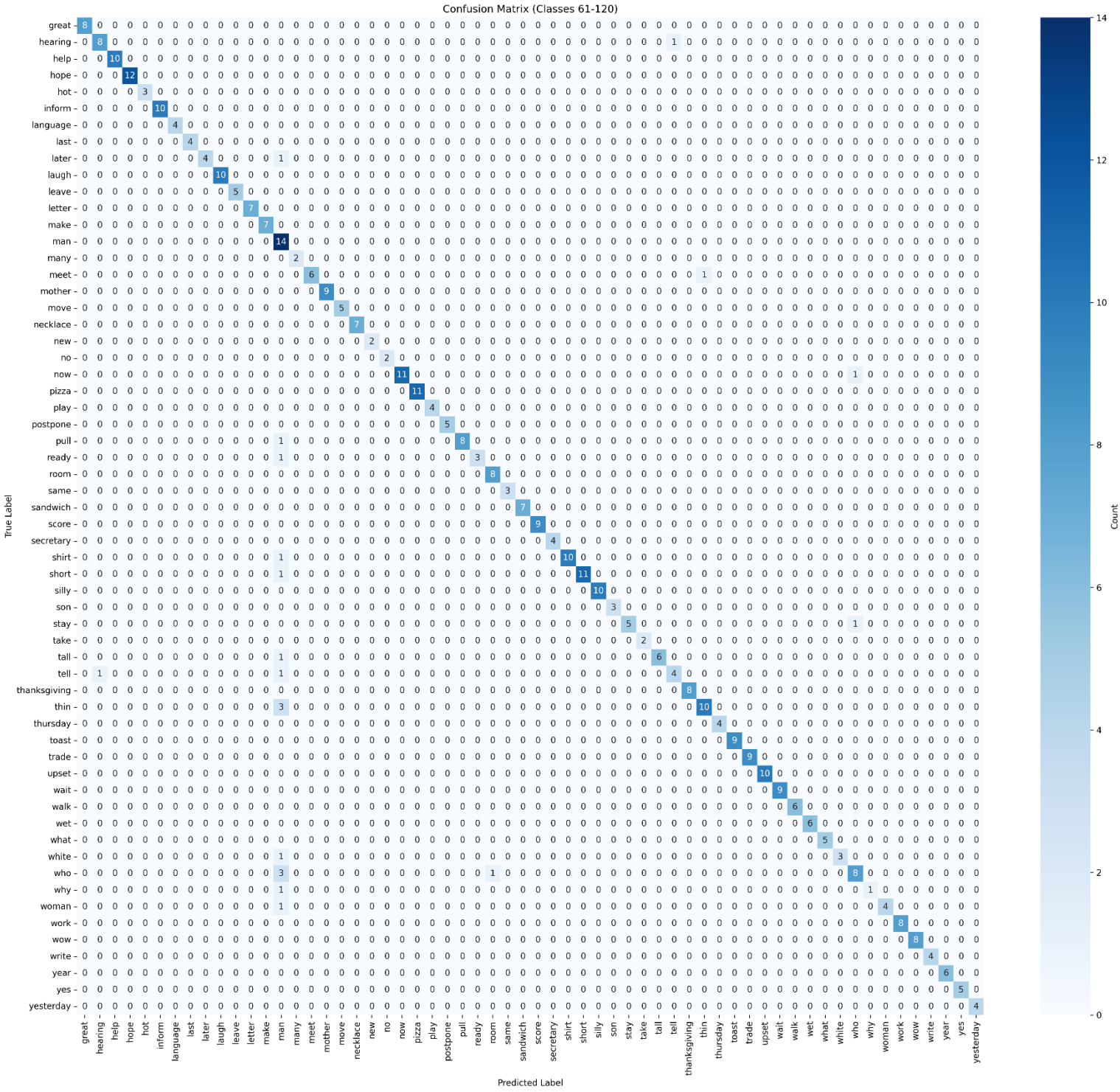


Figure 27: [Confusion Matrix 61-120] (confusion_matrix_61-120.png)

When compared to similar gesture recognition systems, such as basic CNN classifiers or 3D CNNs (Zhao et al., 2023), this system achieves an average performance with far lower computational demand. The 95.48% validation accuracy across the 120 ASL classes indicates the model's practical feasibility. While a few gestures show lower performance due

to class imbalance, this trend is consistent with findings in other sign language datasets where long-tail distribution is common.

6.3.3 Comparative Analysis

When compared to other gesture recognition models, such as those using traditional CNNs or 3D CNNs on datasets like MS-ASL or WLASL, the proposed ConvLSTM2D model offers a balance of performance and efficiency. While 3D CNNs achieve comparable accuracy, they are computationally intensive and less suited for deployment in browser-based systems. In contrast, the developed model achieves a validation accuracy of 95.48% across 120 ASL classes with a reduced inference overhead. This makes it ideal for real-time applications where responsiveness and resource limitations are key concerns. These findings align with research indicating that temporal modeling through recurrent layers can enhance video-based gesture classification without the full cost of volumetric models (Shi et al., 2015; Zhao et al., 2023).

6.4 Reflection on Outcome

The project demonstrates the feasibility of building a real-time, web-based sign language interpreter using lightweight AI models. The integration of ConvLSTM2D with MediaPipe provides a cost-effective solution capable of processing spatio-temporal gestures directly in the browser using standard webcams. One of the key achievements is the system's scalability, achieved through modular architecture and the use of Python-based APIs.

Despite strong results, several areas emerge for improvement. The system struggles with a few visually similar gestures and underrepresented signs, revealing the need for dataset balancing and more advanced temporal modeling. The experience highlights the critical balance between accuracy and latency, especially for accessible systems intended for broad deployment.

The recommendations include incorporating transformer-based temporal decoders, expanding dataset coverage and involving real users for continuous learning via feedback loops. This project also underlines the importance of ethically mindful AI development, particularly when serving marginalized communities.

Chapter 7 – Conclusion and Future Works

7.1 Project Analysis

This report outlines the design, development and evaluation of an AI-driven sign language interpretation system. Deep learning, specifically a ConvLSTM2D architecture, is used to interpret and classify sign gestures. Throughout the development, the primary objective is to make communication more accessible and inclusive for deaf and hard-of-hearing users. Gesture sequences are processed using MediaPipe-extracted hand and face landmarks, and a strong validation accuracy of 95.48% is achieved. From the start, the system is designed to address limitations seen in existing solutions, such as the reliance on specialized sensors or support for only a narrow vocabulary. A solution is delivered that performs with both speed and accuracy. Isolated gesture recognition is effectively supported and a responsive, intuitive web interface is created. As the system is developed, the training setup is iteratively refined, datasets are cleaned and the user interface is enhanced.

7.2 Limitations of Project

While the project achieved good results, several limitations are encountered:

1. Limited Dataset Size and Diversity: Despite using a cleaned version of WLASL/MS-ASL, some classes contain very few examples, leading to poor generalisation for those signs. The model sometimes confuses signs with subtle differences in hand position or motion trajectory.

2. Lack of CSLR: The current system interprets isolated gestures within a 3-second window. Continuous or sentence-level recognition, which is more reflective of real communication, was beyond the project scope due to time and computational constraints.

3. Minimal User Testing with the Deaf Community: While the system is designed to be accessible, limited time and outreach allowed for only a few informal feedback that could be collected. Full-scale usability testing would require more structured collaboration with deaf and hard-of-hearing users.

4. Performance Dependency on Environmental Factors: Background clutter, lighting changes and camera angles affect the consistency of landmark detection and model performance. Although normalization and noise augmentation are applied, a real-world deployment would still require adaptive error handling mechanisms.

5. Real-Time Constraints and Latency: Though prediction speed is optimized, occasional lag occurred during high-load sessions, especially on lower-end devices. More extensive profiling and lightweight deployment strategies such as model quantization or ONNX conversion are needed.

These limitations inform the future direction of work, particularly in enhancing scalability, robustness and inclusivity.

7.3 Future Work

CSLR and translation have become significant due to innovations in deep learning, particularly through multi-modal fusion techniques, Transformer-based temporal modeling and generative adversarial networks. In spite of these advances, a few challenges remain. By merging insights from the studies conducted, our future research directions can be categorized into the following five key areas:

7.3.1 Vocabulary Expansion

Vocabulary expansion is critical for real-world utility. Current systems focus on a limited set of gestures (such as the ASL alphabet or subsets of the MS-ASL dataset); a development is needed to fine-tune larger models on bigger and diverse datasets. This will enhance coverage of a wider vocabulary and inclusion of nuanced gestures and expressions, thereby improving real-world applicability and inclusivity. Consequently, AI-driven sign language systems would become more robust for different linguistic contexts.

7.3.2 Multi-Modal Fusion

The integration of multiple data modalities constitutes a crucial space for improving recognition accuracy, especially in noisy environments. In future research, a planned research may include:

- Combine skeletal key-point extraction, optical flow analysis and depth sensors to integrate all useful visual cues.
- Fuse the non-manual cues (facial expressions and body posture) with hand and arm motion modeling.
- Explore fusion strategies in various layers of the network (early, mid and late fusion) to dynamically weight the importance of each modality based on situational conditions.

7.3.3 Advanced Temporal Modeling

Sign language has a temporal quality, with fluid movement and overlapping gestures. Future systems should therefore integrate:

- Transformer-based architectures or attention-enhanced recurrent networks to capture long-term dependencies for improved segmentation and interpretation of continuous signing.
- Sequence learning modules, with which further experimentation using Conformer models and cross-modal relative attention could be performed, as evidenced by the work so far, to improve alignment between visual features and glosses.
- Finally, integrate unsupervised pretraining techniques to gain contextual learning, thus improving performance on cumbersome benchmarks such as Phoenix-2014 and Phoenix-2014T.

7.3.4 User-Centric Evaluations

The sign language interpreter should, however, primarily be at the service of the deaf community in order to be more functional:

- Extensive usability studies conducted with deaf and hard-of-hearing users to establish practical challenges.
- Regular feedback from users would allow an iterative refinement of the system's performance based on responsiveness, ease-of-use and cultural appropriateness.
- Evaluation of fairness and robustness across multiple signers and demographic backgrounds.

7.3.5 Adaptation and Expansion across Languages

Whereas most currently established systems are designed with ASL in mind, huge possibilities have been opened for adapting these models to the global arena:

- Domain adaptation and transfer learning techniques to develop models applicable to other sign languages.
- Use of common datasets from particular communities (for example, AUTSL for Turkish, Auslan for Australian Sign Language) to enable more inclusive and culturally appropriate systems.
- The proposed fusion approaches are beyond the pure motivation of the two modalities and a consideration of a graph-based cross-modal information fusion would widen the underlying system use and better capture the semantic mapping between different sign language modalities.

7.3.6 Data Augmentation for Generalization

To improve model generalization across varied signing styles and environmental conditions, future work will explore the application of data augmentation techniques. Techniques such as horizontal flipping, rotation, temporal jittering (adding or removing frames) and synthetic noise injection can be used to artificially expand the training dataset. These augmentations can help to enhance robustness to occlusions, camera angles and inconsistent lighting. Moreover, adaptive frame skipping and gesture speed normalization can be investigated to mitigate issues caused by signer variability and motion blur. Implementing such augmentations will likely improve real-world accuracy, especially in uncontrolled, user-submitted videos.

By pursuing these integrated research directions, the AI interpreter systems achieve higher accuracy, robustness and diverse real-world applications. An integrated approach, such as through vocabulary, multi-modal integration, temporal dynamic aspects, user feedback and cross-lingual aspects, opens up possibilities for filling in the gaps while promoting cultural inclusivity. This project has provided a valuable opportunity to explore real-time sign language recognition through AI. A functional prototype has been delivered while challenges have been encountered and limitations identified.

Bibliography

Alyami, S. and Luqman, H. (2024) *'A Comparative Study of Continuous Sign Language Recognition Techniques'*. Available at: https://arxiv.org/abs/2406.12369 (Accessed: 15 January 2025).

Cerna, L.R., Escobedo Cardenas, E.J., Miranda, D.G., Menotti, D. and Camara-Chavez, G. (2021) *A multimodal LIBRAS-UFOP Brazilian sign language dataset of minimal pairs using a Microsoft Kinect sensor*. Available at: https://arxiv.org/abs/2008.00932 (Accessed: 01 February 2025).

Hou, J., Liu, Y., and Wu, Y. (2019) *Watch what you sign: Real-time sign language detection on smartwatches.* Available at: https://dl.acm.org/doi/10.1145/3300061.3300109 (Accessed: 01 February 2025).

Papastratis, I., Chatzikonstantinou, C., Konstantinidis, D., Dimitropoulos, K. and Daras, P. (2021) *Artificial Intelligence Technologies for Sign Language*. Available at: https://www.mdpi.com/1424-8220/21/17/5843 (Accessed: 07 February 2025).

Papastratis, I., Theodorakis, S., and Maragos, P. (2020) *Adversarial training for sign language translation.* Available at: https://ieeexplore.ieee.org/document/9191375 (Accessed: 08 February 2025).

Papastratis, I., Theodorakis, S., and Maragos, P. (2021) *Cross-modal alignment for sign language translation: A case study on Greek sign language*. Available at: https://ieeexplore.ieee.org/document/9414275 (Accessed: 01 February 2025).

Meng, L. and Li, R. (2020) *An Attention-Enhanced Multi-Scale and Dual Sign Language Recognition Network Based on a Graph Convolution Network*. Available at: https://doi.org/10.3390/s21041120 (Accessed: 26 January 2025).

Microsoft (2019) *MS-ASL American Sign Language Dataset*. Available at: https://microsoft.github.io/data-for-society/dataset?d=MS-ASL-American-Sign-Language-Dataset (Accessed: 18 January 2025).

Tavella, F., Galata, A. and Cangelosi, A. (2024) *Bridging the Communication Gap: Artificial Agents Learning Sign Language through Imitation*. Available at: https://arxiv.org/abs/2406.10043 (Accessed: 26 January 2025).

References

Aboaf, E. (2024) '*Challenges in AI for Sign Language: Ensuring Cultural Sensitivity and Robustness, Le Monde'*. Available at: https://www.lemonde.fr/pixels/article/2024/07/31/les-ia-generatives-peuvent-elles-etre-utiles-aux-personnes-sourdes-et-malentendantes_6263205_4408996.html (Accessed: 28 January 2025).

Adaloglou, N., Stergioulas, A., Kouris, A., Theodorakis, S., Giannakopoulos, T., and Maragos, P. (2021) *A comprehensive study on sign language recognition methods. IEEE Transactions on Pattern Analysis and Machine Intelligence*. Available at: https://ieeexplore.ieee.org/document/9675883 (Accessed: 07 February 2025).

Aloysius, N., Geetha, M., Nedungadi, P. (2024) *'Continuous Sign Language Recognition with Adapted Conformer via Unsupervised Pretraining'*. Available at: https://arxiv.org/abs/2405.12018 (Accessed: 25 January 2025).

Avina, V.D., Amiruzzaman, M., Amiruzzaman, S., Ngo, L.B. and Dewan, M.A.A. (2023) *An AI-Based Framework for Translating American Sign Language to English and Vice Versa*. Available at: https://doi.org/10.3390/info14100569 (Accessed: 25 January 2025).

Baskoro, R. (2024) *WLASL-Processed Dataset*, Kaggle. Available at: https://www.kaggle.com/datasets/risangbaskoro/wlasl-processed (Accessed: 21 February 2025).

Bragg, D., Koller, O., Bellard, M. et al. (2019) *Sign language recognition, generation and translation: An interdisciplinary perspective*, Proceedings of the 21st International ACM SIGACCESS Conference on Computers and Accessibility (ASSETS). Available at: https://dl.acm.org/doi/pdf/10.1145/3308561.3353774 (Accessed: 25 January 2025).

Camgoz, N. C., Koller, O., Hadfield, S., & Bowden, R. (2018). *Neural Sign Language Translation, In* Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition *(CVPR).* Available at: https://ieeexplore.ieee.org/document/8578590 (Accessed: 08 February 2025).

Chen, H. et al. (2024) *SignVTCL: Multi-Modal Continuous Sign Language Recognition Enhanced by Visual-Textual Contrastive Learning*. Available at: https://arxiv.org/abs/2401.11847 (Accessed: 26 January 2025).

Chen, Y., Zhang, X. & Li, L. (2019) *Real-time sign language recognition with deep learning: A review*. Available at: https://ieeexplore.ieee.org/document/8744292 (Accessed: 10 February 2025).

Dimitropoulos, K., Daras, P. et al. (2021) *Deep Learning Methods for Sign Language Translation*, *Sensors*. Available at: https://www.mdpi.com/1424-8220/21/7/2437 (Accessed: 25 January 2025).

Emmorey, K. (2002) *Language, cognition and the brain: Insights from sign language research*. Available at: https://staibabussalamsula.ac.id/wp-content/uploads/2024/03/LANGUAGE-COGNITION-THE-BRAIN-staibabussalamsula.ac_.id_.pdf (Accessed: 11 February 2025).

Forster, J., Schmidt, C., Hoyoux, T., Koller, O., Zelle, U., and Ney, H. (2014*) Extensions of the sign language recognition and translation corpus RWTH-PHOENIX-Weather*. Available at: https://www.aclweb.org/anthology/L14-1290/ (Accessed: 01 February 2025).

Galea, C. and Smeaton, A.F. (2019) *Multimodal sign language recognition using deep learning.* Available at: https://dl.acm.org/doi/10.1145/3311823.3311874 (Accessed: 01 February 2025).

Goodfellow, I., Bengio, Y. and Courville, A. (2016) *Deep Learning*, Cambridge. Available at: https://www.deeplearningbook.org/ (Accessed: 11 February 2025).

Halvardsson, G., Peterson, J., Soto-Valero, C. and Baudry, B. (2020) *Interpretation of Swedish Sign Language using Convolutional Neural Networks and Transfer Learning*. Available at: https://arxiv.org/abs/2010.07827 (Accessed: 25 January 2025).

Jaffe, D.L. (1994) *Evolution of Mechanical Fingerspelling Hands for People Who Are Deaf-Blind*, Journal of Rehabilitation Research and Development. Available at: https://www.researchgate.net/publication/234456567_Evolution_of_Mechanical_Fingerspelling_Hands_for_People_Who_Are_Deaf-Blind (Accessed: 24 January 2025).

Kumar, R., Sinha, A., Bajpai, A. and Singh, S.K. (2023) *A Comparative Analysis of Techniques and Algorithms for Recognising Sign Language*. Available at: https://arxiv.org/abs/2305.13941 (Accessed: 26 January 2025).

Koller, O., Forster, J., & Ney, H. (2015). *Continuous Sign Language Recognition: Towards Large Vocabulary Statistical Recognition Systems Handling Multiple Signers*, *Computer Vision and Image Understanding*. Available at: https://www.sciencedirect.com/science/article/abs/pii/S1077314215000533?via%3Dihub (Accessed: 26 January 2025).

Ladd, P. (2003) *Understanding Deaf culture: In search of Deafhood*, *Clevedon: Multilingual Matters*. Available at: https://www.multilingual-matters.com/page/detail/?k=9781853595455 (Accessed: 11 February 2025).

Li, D., Rodriguez, C., Yu, X. and Li, H. (2020) *Word-Level Deep Sign Language Recognition from Video: A New Large-Scale Dataset and Methods Compariso*. Available at: https://ieeexplore.ieee.org/document/9093445/citations#citations (Accessed: 26 January 2025).

Mindbowser. (n.d) *Step by step process of Agile Scrum methodology*. Available at: https://www.mindbowser.com/step-by-step-process-of-agile-scrum-methodology/ (Accessed: 01 February 2025).

Mittal, G., Goyal, P., and Kaur, S. (2019) *Sign language recognition using Leap Motion sensor with machine learning*. Available at: https://www.sciencedirect.com/science/article/pii/S1877050919306623?via%3Dihub (Accessed: 01 February 2025).

Orovwode, H., Oduntan, I.D. and Abubakar, J.A. (2023) *'Development of a Sign Language Recognition System Using Machine Learning'*, in 2023 International Conference on Artificial Intelligence, Big Data, Computing and Data Communication Systems. Available at: https://ieeexplore.ieee.org (Accessed: 01 February 2025).

Pu, J., Zhou, W., Li, H., and Wang, M. (2016) *Sign language recognition with multi-modal features*. Available at: https://dl.acm.org/doi/10.1145/2964284.2964319 (Accessed: 07 February 2025).

Russell, S. and Norvig, P. (2021) *Artificial Intelligence: A Modern Approach*, 4th edition. London: Pearson Education. Available at: https://www.pearson.com/en-us/subject-catalog/p/artificial-intelligence-a-modern-approach/P200000003500/9780137505135 (Accessed: 11 February 2025).

ScienceDaily (2024) *'Breaking barriers: Study uses AI to interpret American Sign Language in real-time'*, ScienceDaily. Available at: https://www.sciencedaily.com/releases/2024/12/241216125906.htm (Accessed: 01 February 2025).

Shi, X. et al. (2015). *Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting*. NeurIPS. Available at: https://proceedings.neurips.cc/paper_files/paper/2015/file/07563a3fe3bbe7e3ba84431ad9d055af-Paper.pdf (Accessed: 18 March 2025).

Sincan, O.M. and Keles, H.Y. (2020) *AUTSL: A Large Scale Multi-Modal Turkish Sign Language Dataset and Baseline Methods.* Available at: https://arxiv.org/abs/2008.00932 (Accessed: 01 February 2025).

Sutton-Spence, R. and Woll, B. (1999) *The linguistics of British Sign Language: An introduction*, Cambridge: Cambridge University Press. Available at:

https://assets.cambridge.org/97805216/37183/sample/9780521637183web.pdf (Accessed: 11 February 2025).

University of Surrey (2025) *'SignGPT: AI to Improve Communication for the Deaf and Hard of Hearing'*, Hearing Review, 21 January 2025. Available at: https://www.surrey.ac.uk (Accessed: 01 February 2025).

Vicars, W.G. (2021) *ASL, American Sign Language*. Available at: https://www.lifeprint.com/asl101/pages-layout/lesson1.htm (Accessed: 25 January 2025).

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). *Advances in Neural Information Processing Systems (NeurIPS).* Available at: https://arxiv.org/abs/1706.03762 (Accessed: 27 January 2025).

Zhang, Y. and Jiang, X. (2024) *'Recent Advances on Deep Learning for Sign Language Recognition'*, Computer Modeling in Engineering & Sciences. Available at: https://doi.org/10.32604/cmes.2023.045731 (Accessed: 15 January 2025).

Wang, J., Zhou, W., and Li, H. (2020) *A comprehensive survey on deep learning-based sign language recognition*. Available at: https://ieeexplore.ieee.org/document/9133333 (Accessed: 07 February 2025).

World Health Organization. (2024) *Deafness and hearing loss*. World Health Organization. Available at: https://www.who.int/news-room/fact-sheets/detail/deafness-and-hearing-loss (Accessed: 11 February 2025).

Appendices

Appendix I - Research Ethics Screening Form

Do not amend before use

Middlesex University Mauritius

Research Ethics Screening Form for Students

Only for students on taught programmes – e.g., BSc, MSc, MA, LLM etc
NOT for PostGraduate Researchers – e.g., MRes/MPhil/PhD degrees

Middlesex University is concerned with protecting the rights, health, safety, dignity, and privacy of its research participants. It is also concerned with protecting the health, safety, rights, and academic freedom of its students and with safeguarding its own reputation for conducting high quality, ethical research.

This Research Ethics Screening Form will enable students to self-assess and determine whether the research requires ethical review and approval via the Middlesex Online Research Ethics (MORE) form before commencing the study. Supervisors must approve this form after consultation with students.

Student Name:	Theyal Dookhy	Email:	td483@live.mdx.ac.uk
Research project title:	Sign Language Interpreter		
Programme of study/module:	CST 3990		
Supervisor Name:	Mr Karel Veerabudren	Email:	k.veerabudren@mdx.ac.mu

Please answer whether your research/study involves any of the following given below:			
1. [H]ANIMALS or animal parts.	☐ Yes	☑ No	☐ Maybe
2. [M]CELL LINES (established and commercially available cells - biological research).	☐ Yes	☑ No	☐ Maybe
3. [H]CELL CULTURE (Primary: from animal/human cells- biological research).	☐ Yes	☑ No	☐ Maybe
4. [H]CLINICAL Audits or Assessments (e.g. in medical settings).	☐ Yes	☑ No	☐ Maybe
5. [X]CONFLICT of INTEREST or lack of IMPARTIALITY. If unsure see "Code of Practice for Research" (Sec 3.5) at: https://unihub.mdx.ac.uk/study/spotlights/types/research-at-middlesex/research-ethics	☐ Yes	☑ No	☐ Maybe
6. [X]DATA to be used that is not freely available (e.g. secondary data needing permission for access or use).	☐ Yes	☑ No	☐ Maybe
7. [X]DAMAGE (e.g., to precious artefacts or to the environment) or present a significant risk to society).	☐ Yes	☑ No	☐ Maybe
8. [X]EXTERNAL ORGANISATION – research carried out within an external organisation or your reseach is commissioned by a government (or government body).	☐ Yes	☑ No	☐ Maybe
9. [M]FIELDWORK (e.g biological research, ethnography studies).	☐ Yes	☑ No	☐ Maybe
10. [H]GENETICALLTY MODIFIED ORGANISMS (GMOs) (biological research).	☐ Yes	☑ No	☐ Maybe
11. [H]GENE THERAPY including DNA sequenced data (biological research).	☐ Yes	☑ No	☐ Maybe
12. [M]HUMAN PARTICIPANTS – ANONYMOUS Questionnaires (participants not identified or identifiable).	☐ Yes	☑ No	☐ Maybe

13. [X]HUMAN PARTICIPANTS – IDENTIFIABLE (participants are identified or can be identified even with pusedo names or subject coding used): survey questionnaire/ INTERVIEWS / focus groups / experiments / observation studies/ evaluation studies.	☐ Yes	☑ No	☐ Maybe
14. [H]HUMAN TISSUE (e.g., human relevant material, e.g., blood, saliva, urine, breast milk, faecal material).	☐ Yes	☑ No	☐ Maybe
15.[H]ILLEGAL/HARMFUL activities research (e.g., development of technology intended to be used in an illegal/harmful context or to breach security systems, searching the internet for information on highly sensitive topics such as child and extreme pornography, terrorism, use of the DARK WEB, research harmful to national security).	☐ Yes	☑ No	☐ Maybe
16. [X]PERMISSION is required to access premises or research participants.	☐ Yes	☑ No	☐ Maybe
17. [X]PERSONAL DATA PROCESSING (Any activity with data that can directly or indirectly identify a living person). For example data gathered from interviews, databases, digital devices such as mobile phones, social media or internet platforms or apps with or without individuals'/owners' knowledge or consent, and/or could lead to individuals/owners being IDENTIFIED or SPECIAL CATEGORY DATA (GDPR[1]) or CRIMINAL OFFENCE DATA. [1]Special category data (GDPR- Art.9): "personal data revealing racial or ethnic origin, political opinions, religious or philosophical beliefs, or trade union membership, and the processing of genetic data, biometric data for the purpose of uniquely identifying a natural person, data concerning health or data concerning a natural person's sex life or sexual orientation".	☐ Yes	☑ No	☐ Maybe
18.[X]PUBLIC WORKS DOCTORATES: Evidence of permission is required for use of works/artifacts (that are protected by Intellectual Property (IP) Rights, e.g. copyright, design right) in a doctoral critical commentary when the IP in the work/artifactis jointly prepared/produced or is owned by another body	☐ Yes	☑ No	☐ Maybe
19. [H]RISK OF PHYSICAL OR PSYCHOLOGICAL HARM (e.g., TRAVEL to dangerous places in your own country or in a foreign country (see https://www.gov.uk/foreign-travel-advice), research with NGOs/humanitarian groups in conflict/dangerous zones, development of technology/agent/chemical that may be harmful to others, any other foreseeable dangerous risks).	☐ Yes	☑ No	☐ Maybe
20.[X]SECURITY CLEARANCE – required for research.	☐ Yes	☑ No	☐ Maybe
21.[X]SENSITIVE TOPICS (e.g., anything deeply personal and distressing, taboo, intrusive, stigmatising, sexual in nature, potentially dangerous, etc).	☐ Yes	☑ No	☐ Maybe

M – Minimal Risk; X – More than Minimal Risk. H – High Risk

If you have answered 'Yes' to ANY of the items in the table, your application REQUIRES ethical review and approval using the MOREform **BEFORE commencing your research**.

Please apply for ethical approval using the MOREform (https://moreform.mdx.ac.uk/). Consult your supervisor for guidance. Also see *Middlesex Online Research Ethics* (MyLearning area) *and* *www.tiny.cc/mdx-ethics*

If you have answered 'Maybe' to items L, M, X, H then you need to have your application reviewed by your supervisor **BEFORE commencing your research**.

If you have answered 'No' to ALL of the items in the table, your application is Low Risk and you may NOT require ethical review and approval using the MOREform before commencing your research. Your research supervisor will confirm need to confirm this below **BEFORE commencing your research**.

Do not amend before use

Middlesex University Mauritius

Student Signature:..... Date: 10/02/25

To be completed by the supervisor:

Based on the details provided in the self-assesment form, I confirm that:	Insert Y or N
The study is Low Risk and *does not require* ethical review & approval using the MOREform	
The study *requires* ethical review and approval using the MOREform.	

Superivsor Signature:..... Date: 27/02/25

Appendix II - Project Meeting Log

Project Meeting Log CST3990

NAME: Theyal Dookhy

STUDENT NUMBER: M00927617

Date	Comments	Supervisor's signature
23/01	Discussion of Project idea and draft structure of literature review	
18/02	Discussion and review of draft literature review and project development plan	
17/03	Feedback on literature review, Review of final report structure and Discussion on Model development	
02/04	Discussion on wireframes, website implementation, model integration in website and model development	
07/04	Discussion on report structure, model implementation and evaluation part and structure of presentation slides	
10/04	Discussion and review of report and presentation slides	

Appendix III - Website Designs

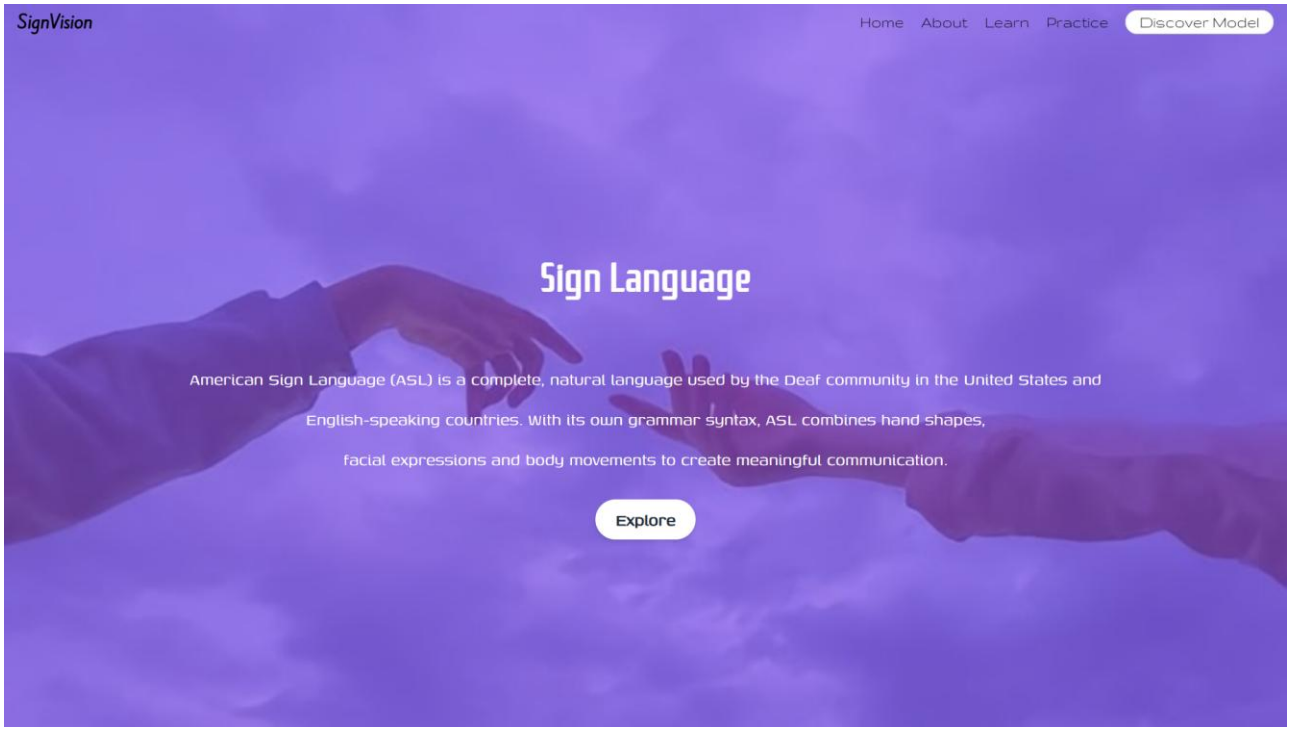


Figure 28: Home Page

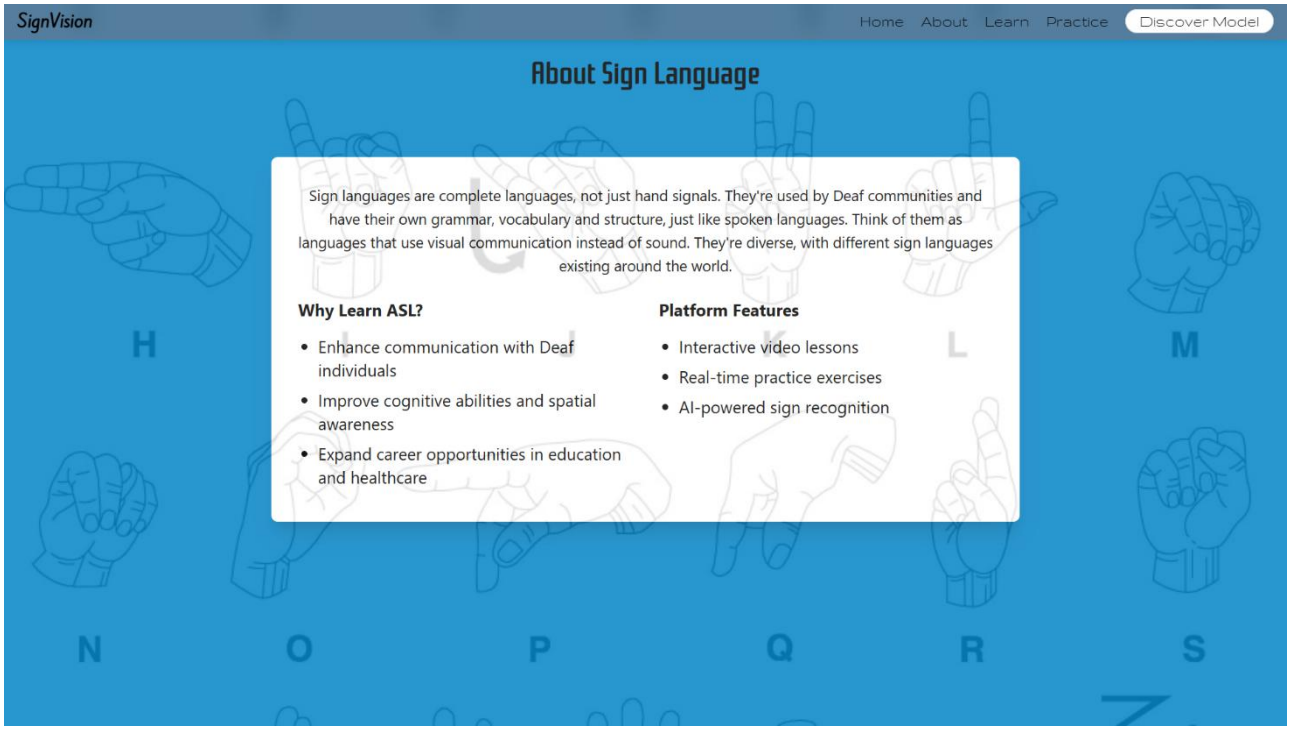


Figure 29: About Page

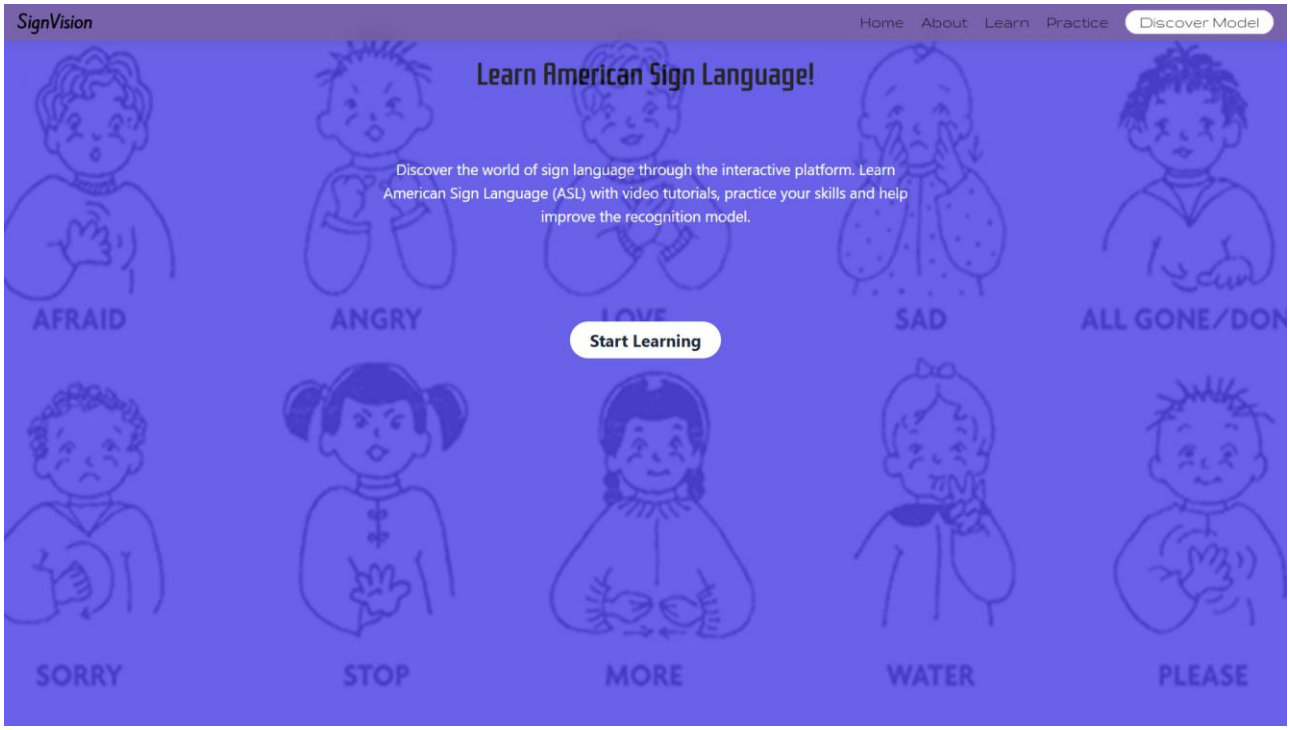


Figure 30: Learning Information Page

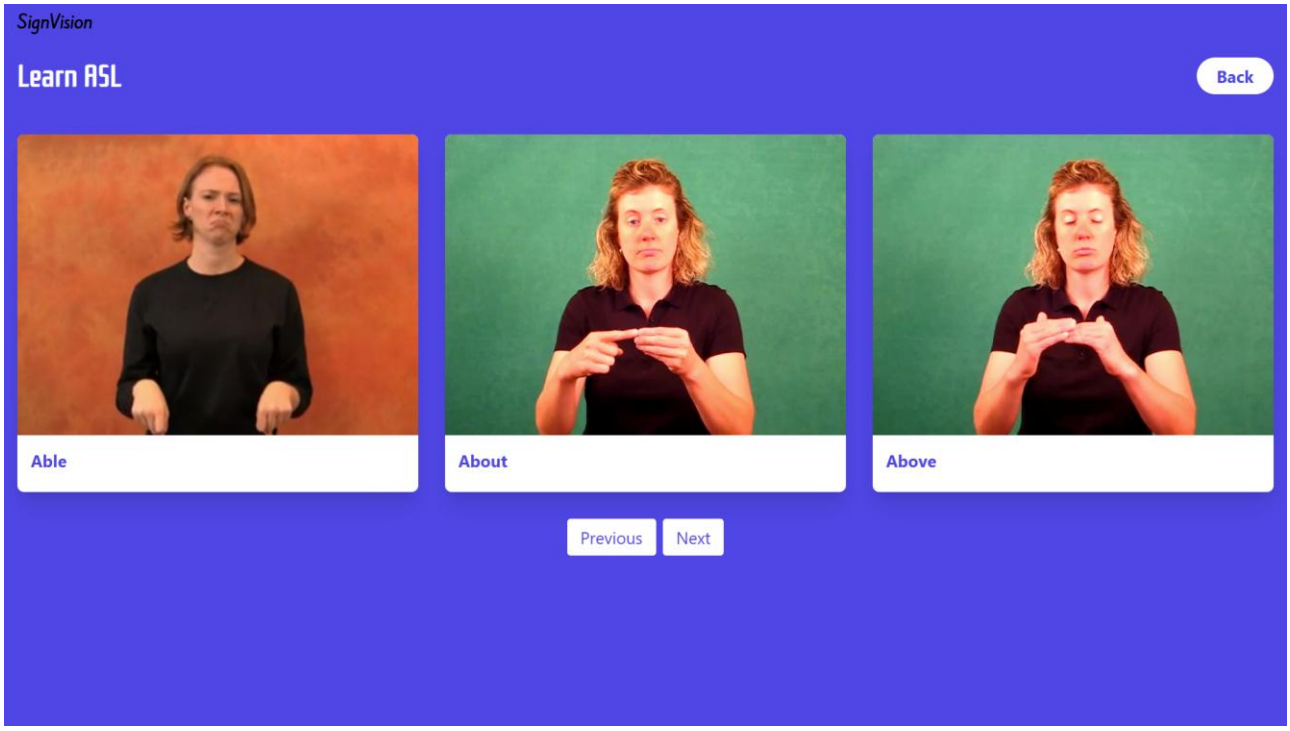


Figure 31: Learning Page

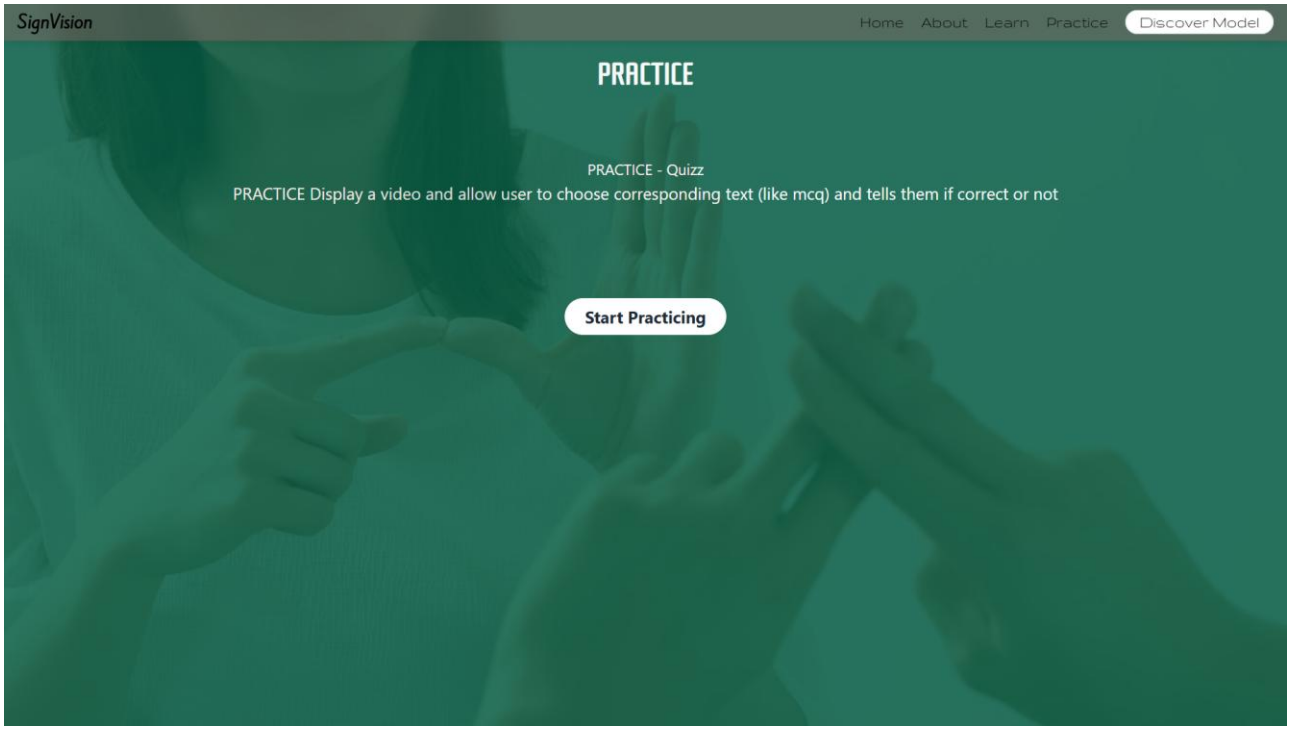


Figure 32: Practice Information Page

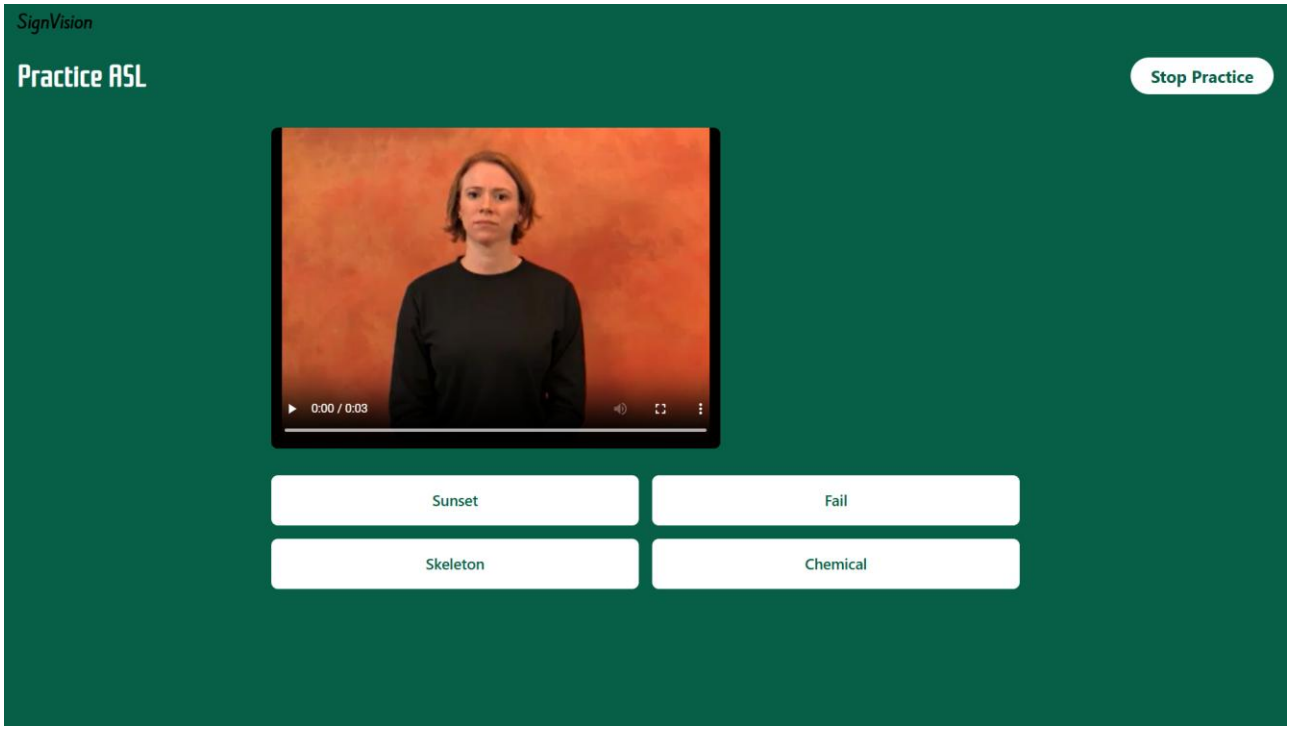


Figure 33: Practice Quiz Page

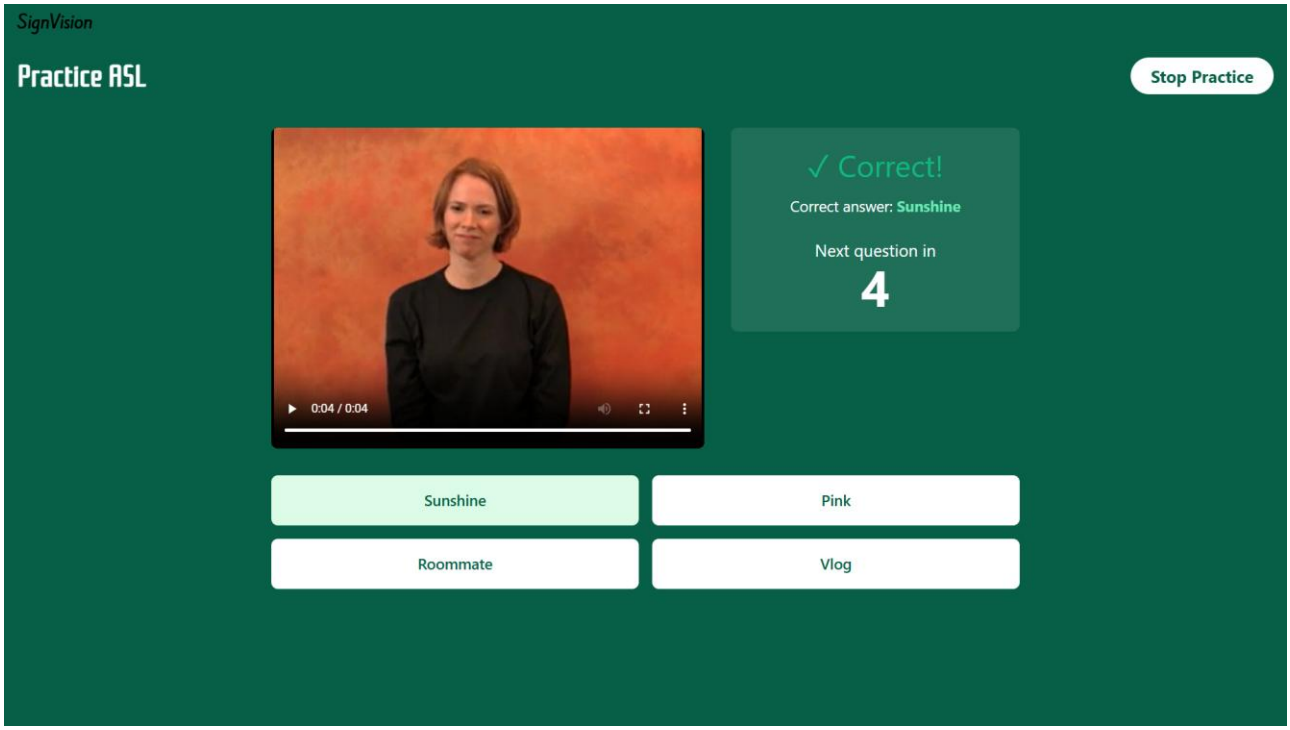


Figure 34: Practice Correct Answer Page

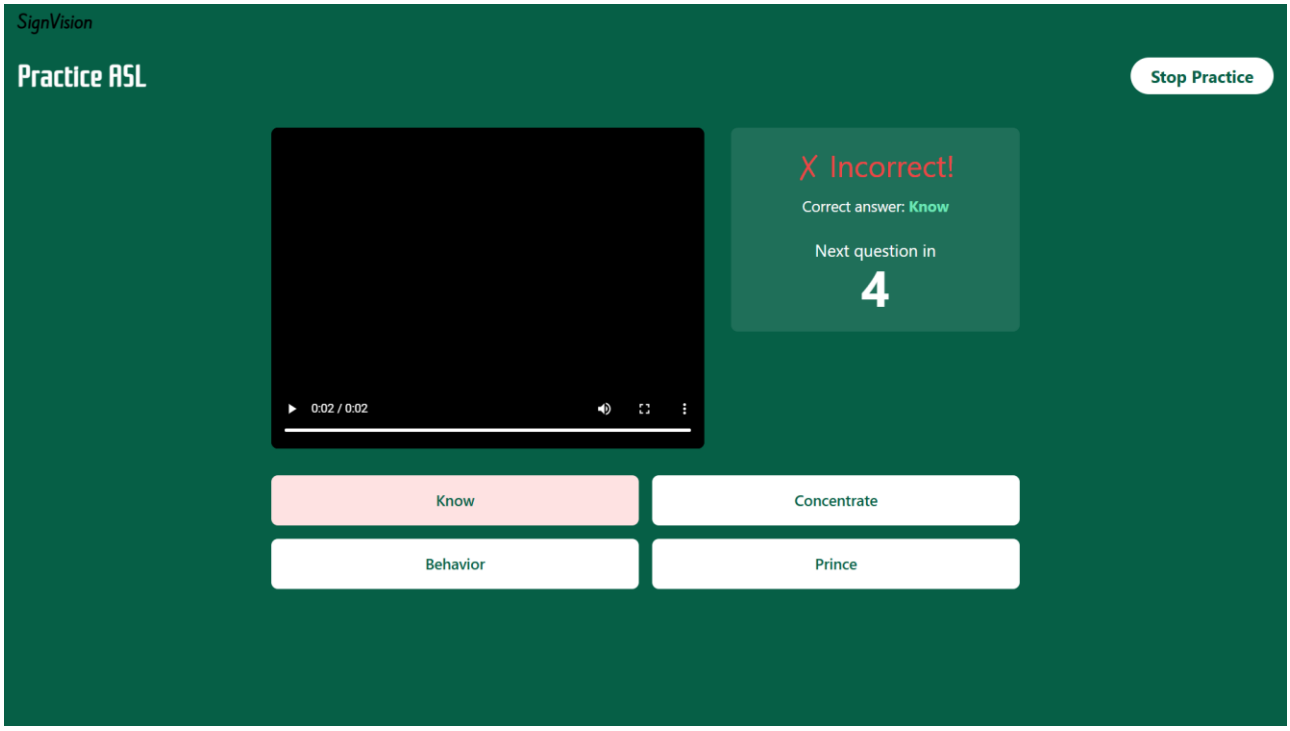


Figure 35: Practice Incorrect Answer Page

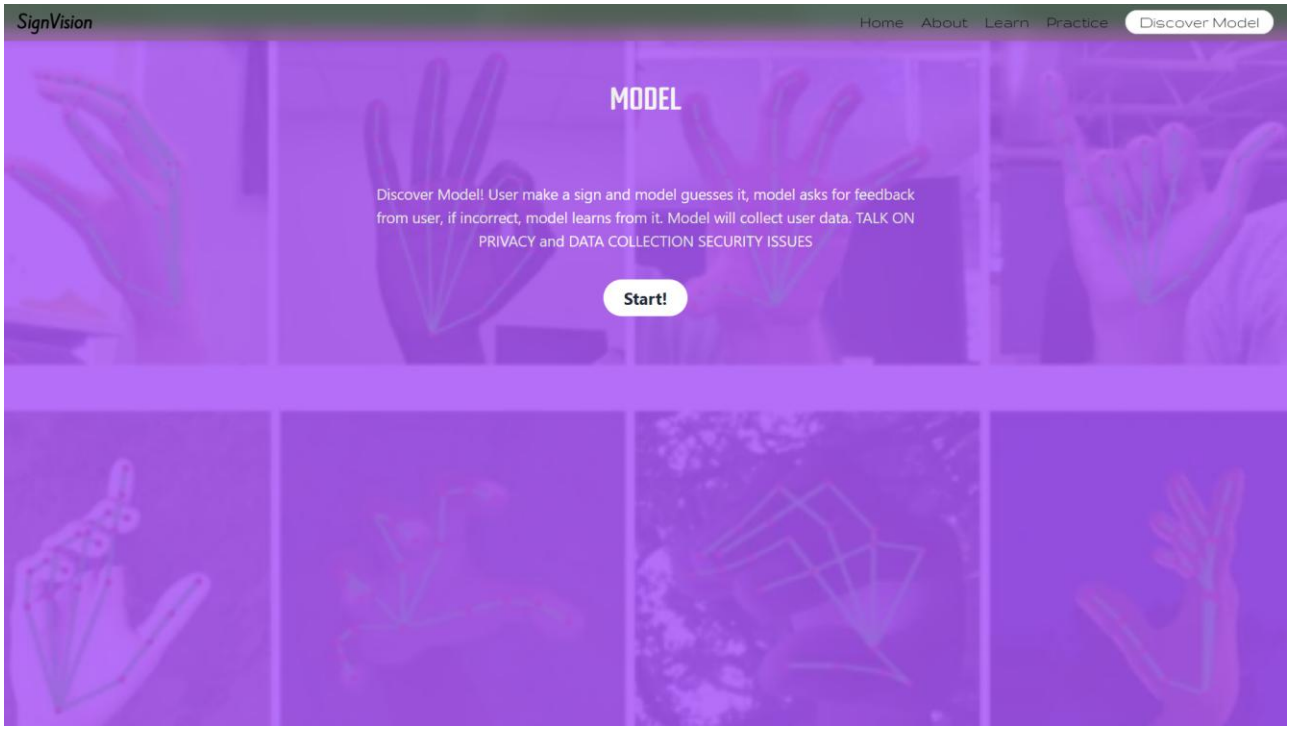


Figure 36: Model Information Page

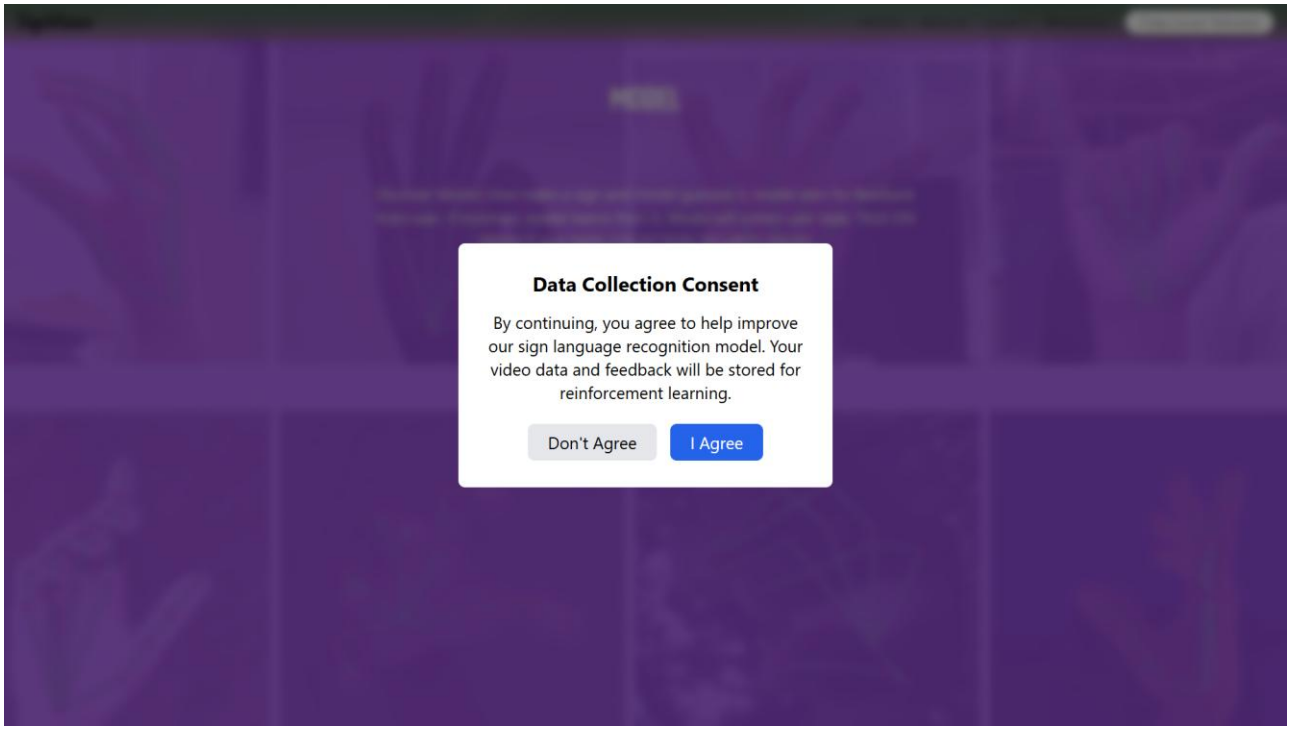


Figure 37: Data Collection Consent Page

Figure 38: Model Feedback Page

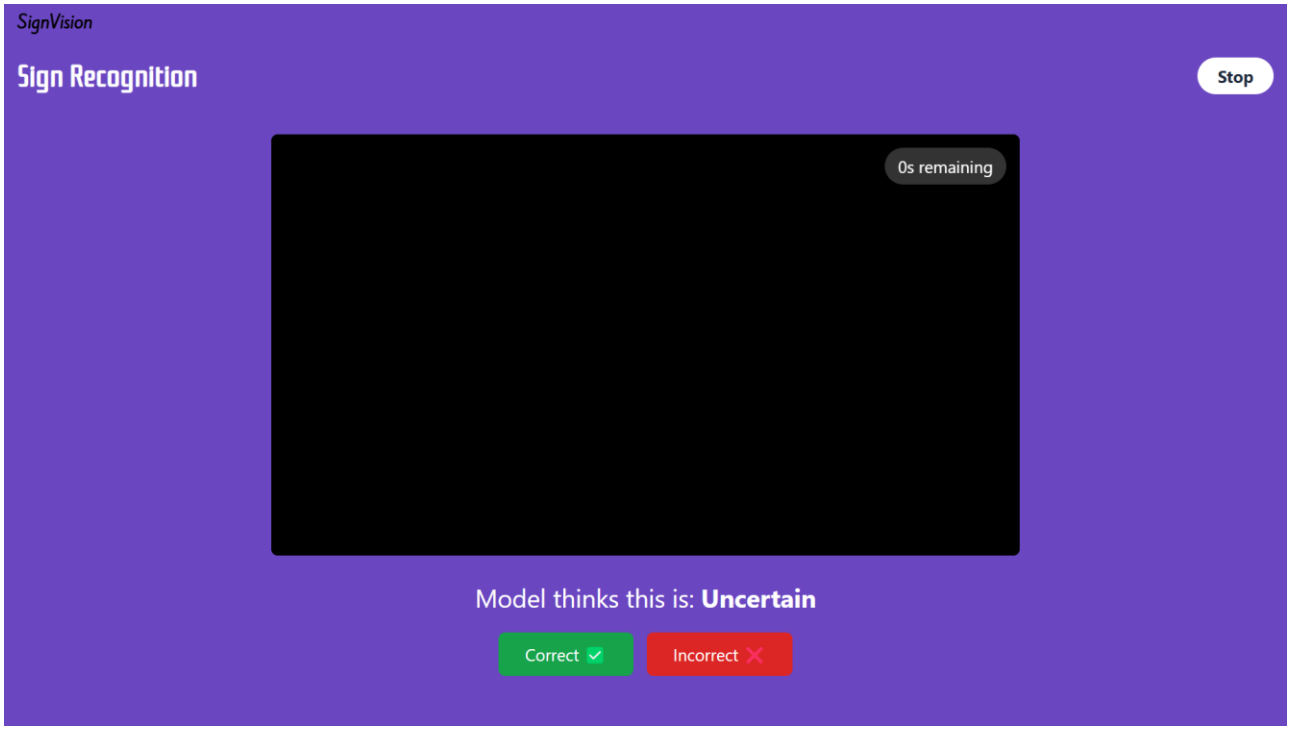


Figure 39: Model Feedback Page(2)