Handly: A Friendly Hand to Help You Understand!

Group 101
Yael Batat (Yael.Batat@mail.huji.ac.il)
Shir Babian (Shir.Babian@mail.huji.ac.il)
Mentored by Gal Katz Hendler

July 30, 2025

GitHub Repository: `https://github.com/YaelBatat/Handly.git`

Abstract

Sign language serves as a crucial communication medium for the deaf and hard-of-hearing community, yet existing learning tools often fail to provide interactive, personalized feedback. To address this gap, we introduce Handly, an intelligent educational software platform designed specifically to teach sign language through real-time, targeted guidance.

Handly leverages MediaPipe's pose estimation to extract skeletal keypoints from user-performed gestures and compares them against reference signs. By employing a K-Nearest Neighbors (KNN) anomaly detection model, the system identifies deviations and classifies gestures as correct or incorrect, offering visual feedback that pinpoints specific movement errors. Unlike conventional video-based tools, Handly features a real-time guided practice mode: the system displays reference videos frame-by-frame, pausing until the user matches each pose accurately, thus enabling step-by-step imitation with immediate correction.

Currently supporting over 1,500 signs, Handly combines computer vision, interactive feedback, and an extensive gesture library to create an accessible and structured learning environment. Our approach not only improves engagement but also addresses the limitations of passive learning methods, making sign language mastery more attainable for learners at all levels.

1 Introduction

1.1 Problem Statement

American Sign Language (ASL) is a visually complex and spatially expressive language, serving as a primary communication tool for the deaf and hard-of-hearing (HoH) community. While digital resources for learning ASL have become more widespread, they overwhelmingly rely on passive instructional formats—typically videos or illustrations—with no mechanism for evaluating user performance. This lack of real-time feedback prevents learners from recognizing and correcting errors in critical components such as handshape, motion trajectory, and palm orientation. Consequently, inaccurate forms are repeated and reinforced, resulting in inefficient learning and limited progress. This absence of interactivity represents a key shortcoming in the current landscape of sign language education.

1.2 Importance of the Problem

High-quality, accessible tools for ASL learning are essential for promoting inclusive communication and reducing barriers between deaf and hearing individuals. Approximately 500,000 people in North America use ASL as their primary language, and worldwide, sign language is used by more than 1.5 billion individuals. Beyond the deaf community itself, many others—including family members, educators, healthcare professionals, and interpreters—seek to learn ASL to better serve or communicate with deaf individuals. However, access to fluent instructors is often limited, and self-study tools without feedback leave learners unable to gauge their progress or correct misunderstandings. Addressing this gap can dramatically enhance the accessibility and effectiveness of ASL education.

1.3 Challenges and Complexity

Developing an effective ASL learning platform involves technical, pedagogical, and practical challenges. Technically, the system must perform accurate, real-time pose estimation across varied users and conditions—handling differences in body types, lighting, backgrounds, and camera angles.

Pedagogically, it must provide feedback that is clear, actionable, and encouraging, helping users understand and correct errors. A central challenge is designing a learning process that leads to real improvement over time—adapting to the learner's pace and focusing on high-impact corrections.

Practically, the platform must run smoothly on standard consumer hardware without requiring specialized sensors or complex setups, ensuring broad accessibility.

1.4 Limitations of Existing Solutions

Existing sign language technologies are primarily built for recognition or translation rather than for education. They convert signs into text or speech but typically lack mechanisms for providing corrective feedback or supporting structured learning. As a result, learners receive no guidance on how their signing differs from the expected standard, making it difficult to improve through self-practice.

1.5 Overview of Our Approach

Handly is an interactive ASL learning platform that combines pose analysis, real-time feedback, and gamification to enable autonomous, structured practice. It is composed of four core components:

- **Sign Practice with Instant Analysis**: After completing a sign, the system presents a side-by-side video comparing the user's performance to a reference. The learner's attempt is color-coded—green for correct, orange for near-accurate, and red for incorrect joints—while a summary label provides immediate status ("Correct" or "Needs Work").
- **Real-Time Guided Practice**: For more challenging signs, Handly decomposes each gesture into sequential frames. The system pauses at each frame until the learner achieves sufficient accuracy, offering live visual overlays, detailed voice, and text feedback to guide improvement.
- **Personalized Games**: Three mini-games—Quiz Mode, Matching Game, and Bubble Shooter—provide targeted repetition of signs that the user previously struggled with.
- **Progress Tracking**: Handly tracks each learner's accuracy history, sign mastery over time, daily practice streaks, and game performance. This enables personalized recommendations, reinforces consistent practice, and helps users monitor their growth visually through a dashboard.

Together, these components create a learner-centered experience that transforms passive content consumption into active skill acquisition.

Our broader vision is to replicate the benefits of working with a human tutor—real-time feedback, personalized correction, and guided practice—through scalable computer vision. By enabling learners to practice anytime and anywhere on any standard webcam-enabled device, Handly has the potential to make ASL education more inclusive, accessible, and effective at scale.

1.6 Summary of Contributions

This work introduces **Handly**, a novel ASL learning platform that combines pose tracking, real-time corrective feedback, personalized gamification ,and progress tracking to support independent sign language acquisition.

1.7 Related Work

Several systems have addressed sign language recognition, but few focus on active learning and feedback for learners. This section reviews key approaches and contrasts them with Handly's learner-centered design.

- **Hand Talk Translator**([1]) Hand Talk Translator is a widely used application that converts spoken Portuguese into sign language using 3D avatars (Hugo and Maya). The platform allows users to select the language (e.g., Libras or ASL), save favorite translations, and customize avatars with outfits and backgrounds. While effective for

real-time communication, its primary function is translation rather than instruction. It lacks mechanisms for user feedback, error correction, or progress tracking, limiting its utility as an educational tool.

- **ASL Recognition with Deep Learning**([2]) This research presents a deep learning-based system that combines convolutional neural networks (CNN) and long short-term memory networks (LSTM) for ASL recognition. It successfully captures both spatial and temporal features from videos, achieving high classification accuracy. However, the system is geared toward recognition tasks and does not support user interaction, error visualization, or structured educational feedback—key components for language acquisition.

- **SignAll**([3]) SignAll is a sophisticated real-time ASL translation system developed by Dolphio Technologies. It integrates multiple camera angles, computer vision, and natural language processing to translate ASL into grammatically coherent text. SignAll supports both Hungarian Sign Language (HSL) and ASL and has been recognized internationally for its technical achievements. Despite its innovation in bidirectional communication, SignAll does not offer features tailored to learners, such as feedback loops, performance tracking, or corrective guidance.

- **NVIDIA Maxine Sign Language AI**([4]) NVIDIA's Sign Language AI is a platform that supports ASL learning by allowing users to contribute videos and receive real-time recognition feedback. Its goal is to build a growing community-driven dataset. Although the system enables scalable engagement, it focuses primarily on the relative positioning of the hands and general sign detection. It does not provide detailed feedback to help users understand the specific nature of their errors or how to improve, which limits its effectiveness for learners requiring guided instruction.

Comparison to Handly Unlike the aforementioned systems, Handly is purpose-built for ASL education. It offers real-time, frame-by-frame feedback, personalized correction overlays, progress tracking, and gamified repetition. Handly shifts the focus from passive recognition to active learning by supporting a structured and self-guided learning environment. Moreover, it directly addresses gaps in earlier systems, such as the lack of feedback on joint-level accuracy, adaptive timing adjustments, and sign mastery analytics.

2 Approach, Methods & Materials

2.1 Solution Pipeline Overview

Handly is an interactive web-based platform for learning American Sign Language (ASL), designed to provide real-time, personalized feedback through a combination of computer vision, machine learning, and gamified learning. The platform is structured into four primary modules: (1) an initial practice module for uploading or recording user-performed signs and receiving immediate feedback, (2) a frame-by-frame guided practice mode, (3) educational games that reinforce difficult signs, and (4) an analytics dashboard to track user progress.

The solution pipeline is composed of three key stages: preprocessing and training, real-time prediction and feedback, and user interaction. These stages are outlined below and visually summarized in the accompanying diagrams.

2.2 Preprocessing and Training Phase

This phase involves preparing the dataset, extracting pose features, and training the model.

1. **Dataset Preparation:** Handly is trained on a curated Kaggle dataset of 1,500 ASL signs, each with at least three reference videos (720p, 30fps).

2. **Pose Detection:** MediaPipe Holistic extracts 33 body and 21 hand landmarks from each frame.

3. **Alignment:** Affine transformations normalize user and reference poses using shoulder, hip, and nose landmarks. To ensure consistent comparisons across users, we aligned all videos to the center of the first frame and scaled them based on shoulder width, compensating for differences in body size, camera distance, and recording angle.

4. **Embedding:** Processed pose sequences are converted into 768-dimensional feature vectors using SignCLIP — a contrastive learning model originally designed for matching sign language poses to textual labels ([5]). In Handly, SignCLIP is used not for classification, but as a feature extractor that generates pose embeddings capturing spatial and temporal structure of the sign.

5. **k-NN Training and Thresholds:** A k-Nearest Neighbors model (k = min(number of videos for this word, 6)) is trained on these embeddings using cosine similarity. For each word, a decision threshold is computed based on intra-class distances among correct examples. These thresholds are stored separately and loaded during prediction to determine classification outcomes.

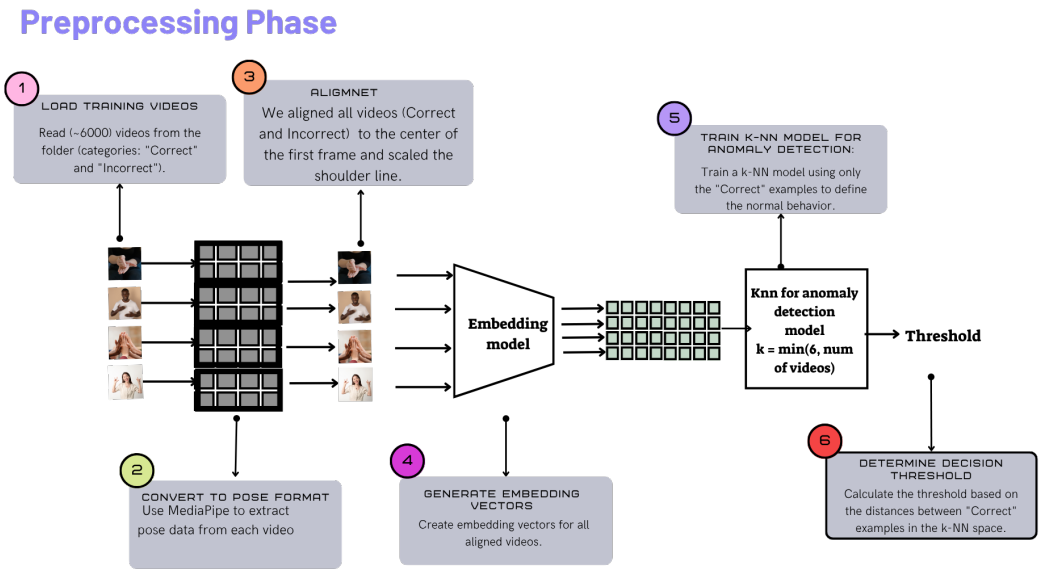

Figure 1: Preprocessing and training workflow

2.3 Prediction and Feedback Phase

This phase describes the internal processing pipeline used by the system to classify user-performed signs and generate feedback.

1. **User Input:** The system accepts either a recorded or uploaded video of the user performing a selected sign. Both types of input follow the same processing pipeline.

2. **Pose Extraction:** The video is processed using MediaPipe Holistic to extract 33 pose and 21 hand landmarks per frame.

3. **Alignment:** The same affine transformation process used during training is applied at inference time to normalize the user video based on anatomical landmarks (shoulders, hips, nose). Each user video is aligned to the center of its first frame and scaled based on shoulder width, ensuring consistency with the already aligned reference videos. This compensates for variations in user body proportions, camera distance, and recording angles, enabling fair and accurate comparison.

4. **Embedding Generation:** The aligned user video and corresponding reference videos are passed through SignCLIP ([5]) to generate 768-dimensional embedding vectors — following the same embedding process used during preprocessing.

5. **Similarity Comparison:** Cosine similarity is used to compare the user's embedding with those of reference examples. The system calculates the mean distance to the k nearest neighbors, where k = min(number of reference videos for the word, 6).

6. **Threshold Evaluation:** The mean similarity distance is compared to a precomputed, word-specific threshold saved during training. These thresholds are dynamically loaded during inference.

7. **Feedback Generation:** If the similarity score exceeds the threshold, the sign is considered correct. Otherwise, the system generates feedback using the following outputs:

   - Numerical similarity score (0–100%)
   - Side-by-side display of the user's video and the closest reference video
   - Color-coded limb overlays: green (accurate), yellow (minor deviation), and red (significant error)

   To compute the color overlays, the system compares corresponding joints from the user's and reference poses using normalized Euclidean distances. Joints with a distance < 0.04 are colored green (excellent match), between 0.04–0.08 yellow (minor error), and > 0.08 red (significant error). This intuitive visual aid helps users quickly identify which parts of their sign require correction.

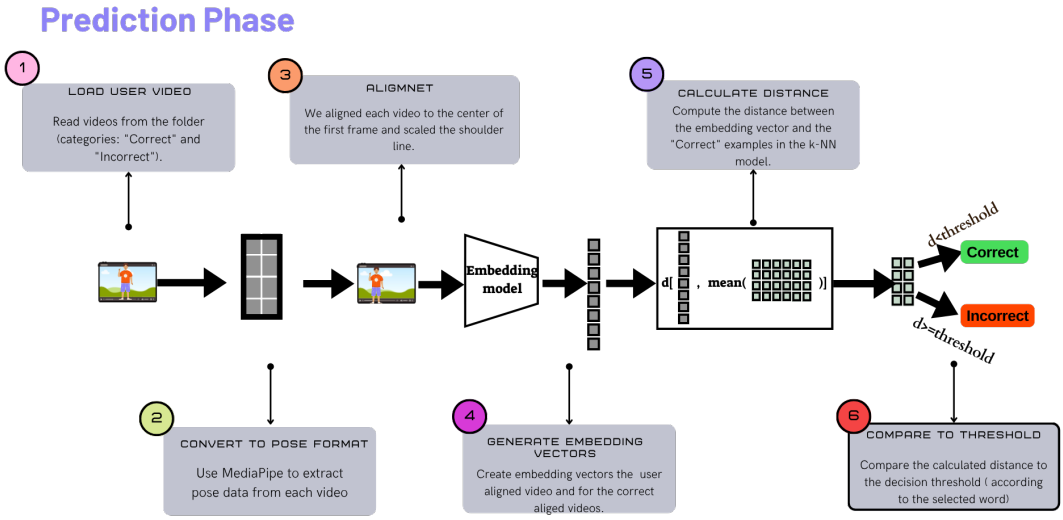

Figure 2: Real-time prediction and feedback generation pipeline

2.4 User Interaction Flow

This section focuses on the user experience, from initial practice to performance tracking.

1. **Initial Practice:** The user begins by selecting a sign to practice—either randomly assigned or by requesting a new random word. Once selected, the system presents a demo video illustrating the correct gesture. Users can adjust the demo's playback speed (e.g., 0.5x, 1x) to better observe and understand the motion.

   After reviewing the demonstration, the user may upload or record a video of themselves imitating the sign. If recording, the system provides setup instructions: stand approximately 2 meters away, ensure full upper-body visibility, and place hands at waist level. To improve input quality, the 3–2–1 countdown only starts after both hands are detected at the waist.

Once the video is submitted, the system analyzes the performance and classifies the sign as either correct or incorrect, using the prediction pipeline described earlier. A progress bar is displayed during analysis, which typically completes within 30 seconds. If the sign is classified as incorrect, the user receives detailed visual feedback, including:

- A numerical similarity score (0–100%)
- A side-by-side video display of the user's attempt and the closest reference
- Color-coded overlay of joint accuracy: green (distance < 0.04), yellow (0.04–0.08), red (> 0.08)

Users can control the playback speed of all videos—the demo, the user's own attempt, and the overlaid comparison—to better examine timing and movement alignment. This helps them pinpoint specific body parts that need adjustment.

Depending on the outcome:

- If the sign is **correct**, the system suggests proceeding to a new word.
- If the sign is **incorrect**, the user can choose to retry the same word or switch to a new one.

2. **Frame-by-Frame Guided Practice:** For signs needing improvement, users can enter a guided mode that breaks the sign into keyframes. The user sees a live webcam view with overlayed pose information and a reference video beside it. A progress bar tracks alignment percentage, and audio/textual cues guide correction. The system only advances to the next frame when similarity exceeds 80% for at least 0.4 seconds. Manual navigation between frames is also available.

3. **Gamified Reinforcement:** Users can play one of three games tailored to signs they struggled with:

   - **Quiz Mode:** Identify the correct label for a sign video
   - **Matching Game:** Pair sign videos with corresponding words
   - **Bubble Shooter:** Pop bubbles labeled with correct translations while the sign video plays

4. **Progress Dashboard:** Displays analytics on user activity, accuracy history, sign mastery, daily streaks, and game performance.

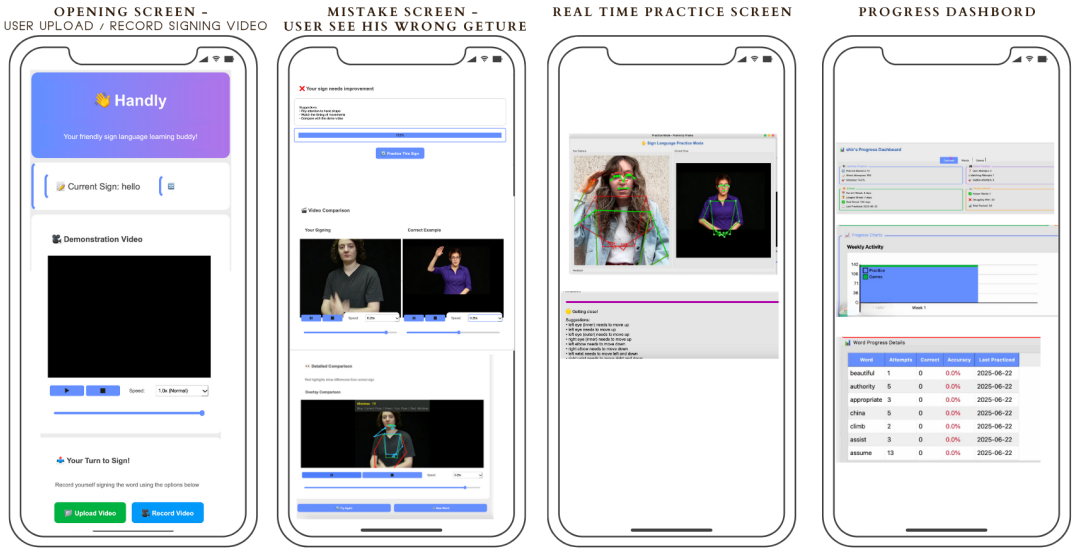

Figure 3: Application user flow: demo, practice, feedback, reinforcement, and progress dashboard

2.5 Technical Methods and Components

Computer Vision Pipeline MediaPipe Holistic is used for pose estimation. Affine transformations normalize scale and orientation. Optical flow and temporal alignment handle gesture timing.

Embedding and Classification We utilize SignCLIP ([5]) as a pose embedding model. SignCLIP was originally designed to project sign videos and corresponding text into a shared embedding space using contrastive learning. In our adaptation, we disabled its text encoder and use only the pose-processing component to generate embeddings for sign comparison. This modification allows SignCLIP to function as a dedicated pose feature extractor within our classification pipeline.

Feedback Mechanism Feedback consists of a similarity score, overlay video with error highlighting, and corrective instructions. This multimodal feedback promotes actionable learning.

2.6 Materials and Resources

Hardware Requirements Webcam (720p minimum), 4GB RAM, optional microphone for audio feedback.

Software Packages Python 3.8, PyQt5 (UI), OpenCV (video), MediaPipe (pose detection), Scikit-learn (k-NN), NumPy (matrix operations), PyGame (game interface), gTTS (Text-to-speech feedback), Google Drive API (storage).

Dataset and Storage 1,500 ASL signs with 3+ reference videos each. Precomputed word-specific thresholds are stored in structured directories on Google Drive. Due to the dataset's size, all media and threshold resources are accessed through Google Drive using the Drive API integrated into the Python backend.

2.7 Implementation Details

The system uses an MVC architecture:

- **Model:** Handles classification, scoring, and data persistence
- **View:** QWidget screens (practice, games, analytics)
- **Controller:** Orchestrates user navigation and system responses

Setup includes dependency installation (requirements.txt) and local or packaged execution. A web version is planned for deployment.

3 Evaluation & Verification

3.1 Verification

We conducted a series of verification procedures to ensure that the system performs its intended functions consistently and reliably.

Embedding Determinism We uploaded the same user video multiple times and confirmed that the resulting 768-dimensional pose embeddings were identical across runs. This confirms the determinism and reproducibility of the embedding pipeline.

Overlay Feedback Accuracy We validated the visual feedback mechanism by uploading both correct and incorrect sign attempts. The system consistently marked misaligned joints in red or yellow and accurate joints in green, according to the defined distance thresholds. Manual inspections across several examples confirmed correct overlay behavior.

Functional Pipeline Verification We verified that each subsystem in the prediction and feedback pipeline operates correctly:

- **Pose Extraction:** MediaPipe consistently extracted 31 body and 22 hand keypoints under varying lighting and backgrounds.
- **Reference Loading:** The correct reference videos and thresholds were dynamically retrieved from Google Drive.
- **Embedding Generation:** Embeddings were computed for every frame without errors, producing valid outputs for classification.
- **Dashboard Updates:** Accuracy history, daily streaks, and game performance metrics updated properly after user activity.

Demonstration Video We created a full-system demonstration video showing a complete user interaction flow: watching a sign demo, submitting a sign attempt, receiving feedback, and seeing progress updates. This serves as a comprehensive qualitative verification of the system's integration and real-time usability (see [6]).

3.2 Evaluation

This section presents a comprehensive evaluation of the system, including quantitative accuracy, systematic self-testing with error analysis, and user testing with real-world participants. We evaluate the robustness of our methods, describe the challenges we encountered, and reflect on limitations that remain unresolved.

Model Accuracy We evaluated the classification performance using a threshold-based approach on six common signs. First, we applied anomaly detection on the correct reference videos for each sign using KNN over their pose embeddings. The value of k for KNN was set as $k = \min(6, N)$, where N is the number of correct samples for the sign ($N \sim$ 3–10 videos). In most cases, this meant $k = 6$, the highest possible for our sample size. This process yielded a similarity-based decision threshold per sign.

We calculated the threshold as the maximum cosine distance observed among the correct examples for each sign, ensuring that all known-good inputs were classified correctly. This empirical threshold helped minimize false negatives.

To perform evaluation, we created a dedicated evaluation directory containing both correct and incorrect examples, including videos we recorded ourselves with varying accuracy levels. For each evaluation video, we computed its cosine distance to the reference set. Based on the threshold, the system classified each video as correct or incorrect, and final accuracy was computed against the ground truth labels.

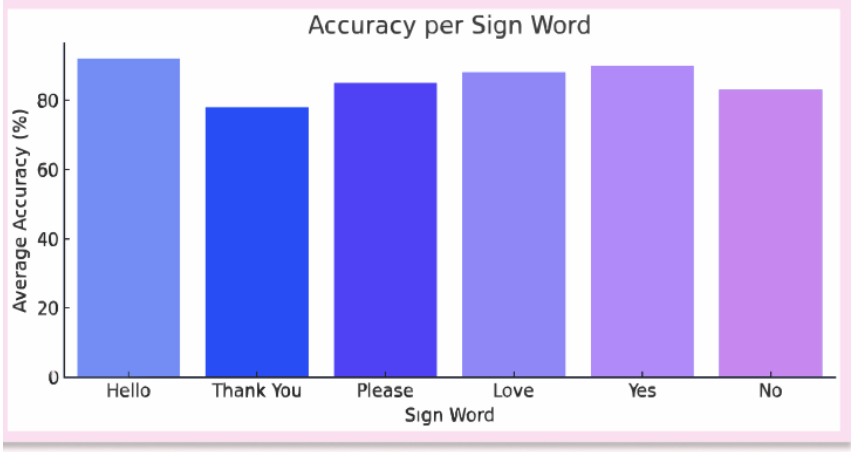

Figure 4: Classification accuracy for six common signs based on evaluation set.

2. Self Testing and Error Analysis We conducted extensive self-testing to probe system sensitivity and uncover limitations under controlled conditions. Self-testing was performed on 35 different signs, with each tested under multiple environmental and user-controlled conditions such as body size, camera angle, and signing speed. This allowed for reproducible analysis of system weaknesses.

Unlike user testing, self-testing enabled us to systematically vary parameters such as camera angle, distance from the camera, signing speed, and sign accuracy in order to observe the system's behavior and limitations in a controlled manner.

2.1 Challenges in Recorded/Uploaded Video Classification Through experimentation, we identified the following challenges:

- **Camera angles:** Signs recorded from oblique angles (>30°) degraded pose accuracy and embedding quality.
- **Body proportions and distance:** Large differences in body shape or camera proximity led to scale and alignment mismatches.
- **Speed and timing variations:** Users often performed signs faster or slower than the reference, with inconsistent start and end timing.
- **Non-standard starts/ends:** Users occasionally began or ended signs with hands outside the frame or in resting positions, confusing the embedding model.

Solutions:

- We applied affine transformations to normalize pose coordinates using shoulder, hip, and nose landmarks, aligning all skeletons to a front-facing canonical view.
- Joint positions were scaled based on shoulder width to account for body size and distance variations.
- We used Dynamic Time Warping (DTW) to align sign sequences based on motion patterns, accommodating different signing speeds.
- A fallback mechanism re-analyzes only the core 90% of a sign if full-sequence classification fails, ignoring unstable start/end frames.

Despite these improvements, one key limitation remains unresolved: the system lacks sensitivity to fine-grained finger positions. In several tested signs, such as those requiring open fingers versus closed fists, the model incorrectly classified incorrect signs as correct if the global hand and arm motion was similar. This is due to the reliance on skeletal pose embeddings, which do not encode finger articulation. This limitation is discussed further in the Future Work section.

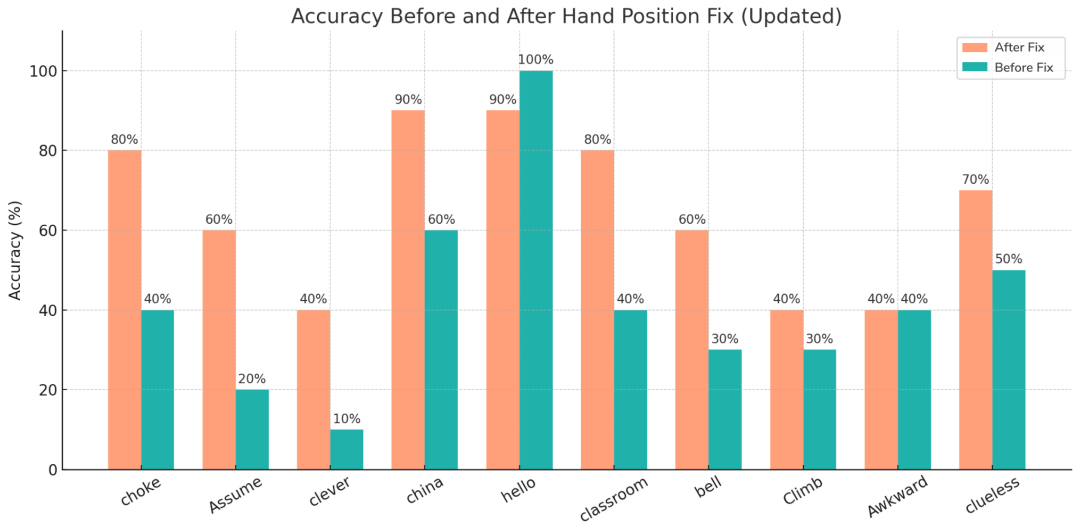

Figure 5: Bar chart showing accuracy for selected signs before and after ignoring unstable start/end frames by reanalyzing only 90% of the sign. Accuracy is computed out of up to 10 tries per word before and after the fix.

2.2 Challenges in Real-Time Guided Practice Real-time frame-by-frame practice introduced a new set of usability and accuracy challenges:

- **Pose overlay mismatch:** Early implementations displayed the reference pose at its original scale and position, making it difficult for users to align their movements to the overlay due to differing body dimensions.
- **Feedback overload:** Users perform rapid movements (0.5–1 seconds per frame), and the system initially generated new feedback for each small change, creating an overwhelming stream of information.

Solutions:

- We aligned the reference pose overlay to the user's skeleton in real time by scaling and translating the joints based on the user's shoulder width and body center. This ensured visual consistency and improved imitation accuracy.
- A color-coded feedback overlay was added: green for accurate joints, yellow for minor errors, and red for significant deviations.
- A similarity bar displaying pose match percentage was introduced to provide continuous feedback.
- To manage feedback flow:
  - Only the last three text feedback messages are displayed.
  - Voice feedback reads the most recent correction and waits until completion before playing the next.
  - Repeated identical messages are suppressed to avoid redundancy.

3. User Testing We conducted real-world user testing with 12 participants aged 8–80 and heights ranging from 120–190cm. Users were given no specific instruction beyond using the system to learn a randomly assigned sign.

Results:

- **Successful recognition** was observed across:
  - Camera angles up to $\pm 30^\circ$ from center
  - Distances of 1–3 meters from the camera
  - Signing speeds between 0.5x–1.5x of the reference
- **Failure cases** occurred when:
  - The camera angle exceeded 45°
  - Multiple people were present in the frame
- Users showed measurable improvement in signing accuracy over time. The integrated progress dashboard helped track individual mastery per sign, and the frame-by-frame feedback mode facilitated self-correction.
- Several users expressed a desire for more engaging learning experiences. In response, we introduced three gamified modes: quiz, matching, and bubble shooter. These reinforced repeated exposure to difficult signs while increasing motivation.

4. Robustness and Sensitivity to Parameters We conducted extensive tuning of key thresholds to balance recognition robustness with user experience. Two components in particular required careful calibration:

- **Fallback Threshold (Core 90% Matching):** If full-sequence classification fails, the system re-analyzes only the core 90% of the sign, ignoring the first and last 5% of frames. We empirically tested thresholds between 90% and 95%, and found that values above 90% were too strict—most failed signs remained misclassified due to minor variations in start or end positions.

  These tests were performed across all 35 signs using a custom test harness that simulated user videos with varied start/end frames. The 90% threshold showed the best tradeoff between accuracy and recovery. Setting this fallback at exactly 90% provided the optimal balance, successfully correcting misclassifications caused by unstable hand positioning at sign boundaries.
- **Frame-by-Frame Progression Threshold:** In guided practice mode, the system advances to the next frame only when the user's pose matches the reference with at least 80% similarity for 0.4 seconds. Lower thresholds (e.g., 70–75%) allowed incorrect or drastically misaligned hand configurations to be accepted—as long as they shared rough body position—leading to poor training quality. Conversely, thresholds above 80% were too rigid, requiring users to precisely match hand height, spacing, and orientation, which hindered progress even when the sign movement was semantically correct. The 80% threshold, paired with a 0.4 second duration, allowed for meaningful pose matching without sacrificing learning flow. It also enabled users to see and feel progress at a comfortable and informative pace.

Summary of Evaluation Our evaluation demonstrated that the system is effective and robust under common real-world conditions. Through self-testing, we resolved key issues related to alignment, scaling, and timing. Real-time feedback was significantly improved through visual, numerical, and auditory cues.

However, one limitation remains: the system's inability to detect fine finger configurations. Signs that differ only in finger articulation may be misclassified as correct. Addressing this limitation requires integrating fine-grained hand modeling or image-based finger detection in future versions.

4 Conclusion & Future Work

Summary of Results Our system demonstrates strong performance in classifying a wide variety of ASL signs using video pose extraction, embedding, and similarity comparison. Through alignment and DTW corrections, we improved accuracy on challenging signs and successfully adapted to varied user inputs.

Limitations and Capabilities The system is robust to variation in posture, distance, and gesture timing. However, it currently lacks sensitivity to subtle finger configurations that are crucial for distinguishing some signs. While MediaPipe Holistic provides coarse hand pose estimation, fine-grained finger articulation is not effectively utilized by the embedding model.

Future Work To significantly improve the system's ability to detect finger positioning errors, we propose enhancing the pose analysis pipeline with fine-grained hand tracking and adaptive scoring. By integrating MediaPipe Hands—which specializes in high-precision finger joint detection—alongside the existing Holistic model, we can capture subtle finger movements that the current system misses.

Next, we plan to modify the similarity scoring to prioritize critical handshapes, assigning higher weights to finger joints (e.g., thumb and index finger positions) while maintaining lower weights for less discriminative body joints. This ensures that incorrect finger configurations—such as closed versus open hands—are penalized more heavily in the final evaluation.

Research Directions and Extensions The success of our visual feedback pipeline opens doors to additional features, such as real-time avatar correction guidance, personalized practice plans based on user error profiles, and domain transfer to other signed languages.

Collaboration and Continuation We envision future collaboration with ASL educators and linguists to refine the system's pedagogical strategies. Additionally, expanding the platform to mobile devices could increase accessibility and reach.

References

1. Hand Talk Translator: `https://www.handtalk.me`

2. ASL Recognition with Deep Learning: https://arxiv.org/abs/2001.09907
3. SignAll: https://www.signall.us
4. NVIDIA Maxine Sign Language AI: https://signs-ai.com
5. SignCLIP GitHub Repository: https://github.com/J22Melody/fairseq/blob/main/examples/MMPT/README.md
6. Demo Video: https://youtu.be/LgLFR__m2aI

Appendix

(Add any supplementary material, e.g., code snippets or additional diagrams.)