American Sign Language Recognition Using Deep Convolutional Architectures

Mohamed Aiman Alkhozendar
mohamed.alkhozendar@bahcesehir.edu.tr

Abstract

This project presents a real-time American Sign Language (ASL) recognition system that translates hand gestures into spoken words using a dual-input Convolutional Neural Network (CNN). My approach integrates both hand images and 3D hand landmarks , improving robustness compared to image-only models. I also experimented several deep learning models—such as custom-built CNNs and pretrained ResNet50 architectures—achieved high validation accuracy using only image inputs, they performed poorly during live webcam inference due to their sensitivity to background variation and lighting. To overcome this, we proposed a dual-branch architecture that fuses image and landmark features. The system confirms predictions based on gesture stability over time, and it uses a text-to-speech engine to vocalize each word upon completion. The final model demonstrates strong real-time accuracy, intuitive user interaction, and practical usability for accessibility applications. My experiments highlight the importance of multimodal inputs for reliable sign language recognition in uncontrolled environments.

1. Introduction

American Sign Language (ASL) serves as a primary communication medium for the Deaf and hard-of-hearing communities. As the world becomes increasingly dependent on digital communication, there is a growing demand for intelligent systems capable of translating sign language into text or speech in real time. This capability can bridge the gap between sign language users and non-signers, enhancing accessibility in education, healthcare, and public services.

The core challenge of ASL recognition lies in its visual complexity and fine-grained hand movements. Traditional computer vision approaches are often limited by variability in lighting, backgrounds, hand sizes, and orientations. Deep learning models—particularly Convolutional Neural Networks (CNNs)—have demonstrated impressive performance on image classification tasks, including hand gesture recognition. However, models trained purely on static images tend to suffer in real-time applications due to their sensitivity to noise and lack of temporal or spatial understanding.

In this project, I developed a real-time ASL translator that leverages a **dual-input CNN architecture**. The model simultaneously processes grayscale images of the hand and 3D hand landmark coordinates extracted using the MediaPipe framework. This multimodal approach enables the model to learn both visual textures and spatial joint structures, improving performance in uncontrolled real-world environments.

My system includes a stability-based prediction mechanism that confirms a gesture only if it is consistently detected for a set duration. Upon confirmation, the letter is added to the sentence, and when the hand is removed from the screen, the system inserts a space and speaks the full sentence aloud using text-to-speech. I further support my approach with extensive experimentation, comparing our dual-input model to custom CNNs and transfer learning architectures such as ResNet50.

The results demonstrate that while image-only models achieve high validation accuracy, their real-world performance under webcam conditions is inconsistent. My final system offers reliable, interactive translation of ASL letters into spoken words in real time, making it practical for real-world use.

2. Related Work

Numerous studies have explored the use of deep learning techniques for sign language recognition, focusing on both static and dynamic hand gesture analysis.

In **Torres and Olvera (2020)** [1], the authors proposed a static ASL recognition system using deep convolutional neural networks trained on grayscale hand images. While their approach achieved promising accuracy in controlled conditions, it lacked robustness when exposed to varying lighting, backgrounds, and hand positions—an issue common among image-only models.

A more comprehensive method was introduced by **Oyedotun and Khashman (2018)** [2], who combined handcrafted features with deep learning to recognize static hand gestures. Their hybrid approach emphasized the importance of integrating spatial and shape-based features into recogni-

tion systems. This aligns with our goal of leveraging structured hand landmark data in addition to visual cues.

Furthermore, **Mittal and Balakrishnan (2021)** [3] investigated real-time hand gesture recognition using MediaPipe and lightweight neural networks. Their study demonstrated that using 3D hand landmarks significantly reduces computational complexity while maintaining accuracy—especially in mobile applications. This supports our decision to use MediaPipe-extracted landmark coordinates as a parallel input stream.

Despite these advancements, few existing works have fused both hand image data and landmark-based spatial features in a unified model. My work builds on this gap by proposing a dual-input CNN architecture that jointly processes image and landmark data to improve real-time accuracy, prediction stability, and user experience.

3. Data Description

3.1. Original Dataset Structure

We used the publicly available *ASL Alphabet Dataset* from Kaggle, which contains over 30,000 RGB images representing 27 classes (A–Z, nothing). For the purpose of this project, we focused on 27 classes: the English alphabet and one additional "Blank" class. Each class folder originally contained approximately 1000 images of size 512×512 pixels.

To support our dual-input model, we restructured the dataset into two synchronized folders:

- **images/**: Cropped and resized grayscale hand images (128×128 pixels)
- **landmarks/**: Corresponding 3D hand landmark arrays extracted using MediaPipe, saved as 21×3 `.npy` files

The folder structure was as follows:

```
/asl_dual_dataset/
 images/
    A/
    B/
    ...
 landmarks/
     A/
     B/
     ...
```

3.2. Data Splitting

The dataset was randomly shuffled and split into:

- 80% training set
- 20% validation set

Care was taken to preserve class balance across both splits.

3.3. Landmark Extraction

I used the MediaPipe Hands framework to extract 21 hand landmarks per image. Each landmark consists of 3 coordinates (x, y, z). The landmarks were normalized by subtracting the mean position across all points, making them invariant to position and scale to some extent. Any images where hands were not detected were discarded during preprocessing.

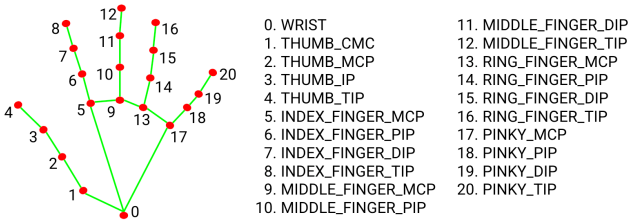

Figure 1. Hand landmarks used

The resulting landmark array per sample was saved as a 21×3 `.npy` file with the same filename as its corresponding image, allowing easy synchronized loading during model training.

3.4. Data Augmentation

To improve generalization and prevent overfitting, we applied moderate augmentation to the image input stream using TensorFlow layers. The transformations included:

- Random rotation ($\pm$15 degrees)
- Random horizontal flip
- Random zoom (up to 10%)
- Random contrast and brightness variation
- Random cropping and translation

However, we deliberately avoided the following:

- **Salt-and-pepper noise:** This introduced unrealistic pixel-level artifacts that do not typically appear in real webcam usage.
- **Strong affine transformations:** These tended to distort hand shapes excessively, degrading model performance.
- **Vertical Flips:**These were considered unrealistic and unlikable to happen due to the hand positioning

The goal was to simulate webcam noise and pose variability without disrupting the structural integrity of the hand signs.

3.5. Additional Dataset Experiments

Throughout the development phase, I conducted extensive experiments using a variety of image sources:

- **Synthetic images:** Artificially generated sign images using flipped, rotated, or digitally altered versions of training samples.
- **Webcam-based images:** Frames were captured in real time using a standard laptop webcam to test performance in uncontrolled lighting and background conditions.
- **Custom dataset:** We manually created a small dataset of ASL gestures using our own hand in various lighting,

angles, and backgrounds to mimic real deployment scenarios.

These experiments helped expose limitations in image-only models and motivated the inclusion of hand landmark coordinates to improve spatial understanding and cross-user robustness.

4. Methodology

4.1. Overview

The goal of this project was to develop a robust and real-time American Sign Language (ASL) alphabet recognition system using deep learning. My primary contribution is a **dual-input Convolutional Neural Network (CNN)** that simultaneously processes grayscale hand images and 3D hand landmarks extracted using MediaPipe.

To achieve this, I designed an end-to-end pipeline that includes: webcam frame acquisition, hand landmark extraction, grayscale cropping and enhancement, data augmentation, dual-input model inference, prediction stabilization, and real-time audio feedback via text-to-speech.

4.2. Initial Model Experiments

Prior to settling on the final architecture, I experimented with:

- **Simple CNNs:** I built lightweight custom CNNs using 2 to 5 convolutional layers with batch normalization, ReLU, and max pooling. These models achieved high validation accuracy but struggled under webcam input due to overfitting to static training conditions.
- **Transfer Learning (ResNet50):** I fine-tuned pretrained ResNet50 models on grayscale ASL images. Although ResNet50 reached over 95% validation accuracy, it was computationally expensive and failed to generalize well in real-time due to sensitivity to background and pose variation.

These results highlighted the need for additional spatial structure input—leading us to the dual-branch architecture.

4.3. Proposed Dual-Input CNN Architecture

Image Branch

- Input: 128×128 grayscale image
- Conv2D (32 filters, 3x3, ReLU) + MaxPooling2D (2x2)
- Conv2D (64 filters, 3x3, ReLU) + MaxPooling2D (2x2)
- Conv2D (128 filters, 3x3, ReLU) + MaxPooling2D (2x2)
- Dropout (rate = 0.3)
- Flatten

Landmark Branch

- Input: 21×3 landmark matrix reshaped to (21, 3, 1)
- Conv2D (32 filters, 3x1, ReLU) + MaxPooling2D (2x1)
- Conv2D (64 filters, 3x1, ReLU)
- Dropout (rate = 0.3)
- Flatten

Fusion and Classification Head The outputs of the two branches are concatenated and passed through a shared classification head

This dual-branch architecture allows the model to simultaneously learn visual texture features from images and spatial joint configurations from landmarks. The fusion improves robustness under varying conditions and supports real-time performance.

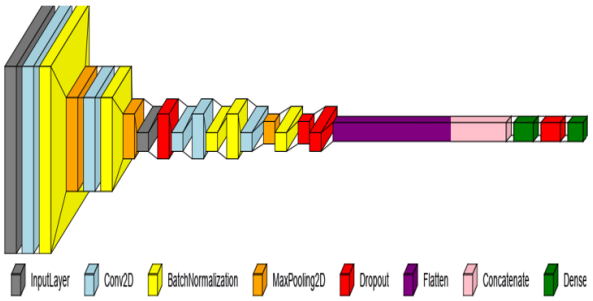

Figure 2. Proposed Dual-Input CNN Architecture: Image and Landmark branches are processed separately and then fused.

4.4. Prediction Stabilization and Speech Output

To ensure reliable real-time classification, we introduced a prediction stabilization mechanism:

- The model only confirms a prediction if the same letter is detected consistently for 1.5 seconds.
- Once confirmed, the letter is added to the sentence string.
- If the hand is completely removed from the screen, a space is inserted and the entire sentence is spoken using the `pyttsx3` text-to-speech engine.

This logic avoids false positives due to transient hand poses or accidental motion, ensuring a smooth user experience.

4.5. Model Training Details

The model was implemented using TensorFlow and trained on Google Colab using an A100 GPU. The following hyperparameters were used:

- Optimizer: Adam
- Learning rate: 0.001
- Batch size: 64
- Loss function: Categorical cross-entropy
- Epochs: 30 (with early stopping based on validation accuracy)

Input images were augmented using random rotation, zoom, horizontal flip, and brightness/contrast changes. Landmarks were augmented using small random noise to simulate hand jitter.

4.6. Pipeline Flow

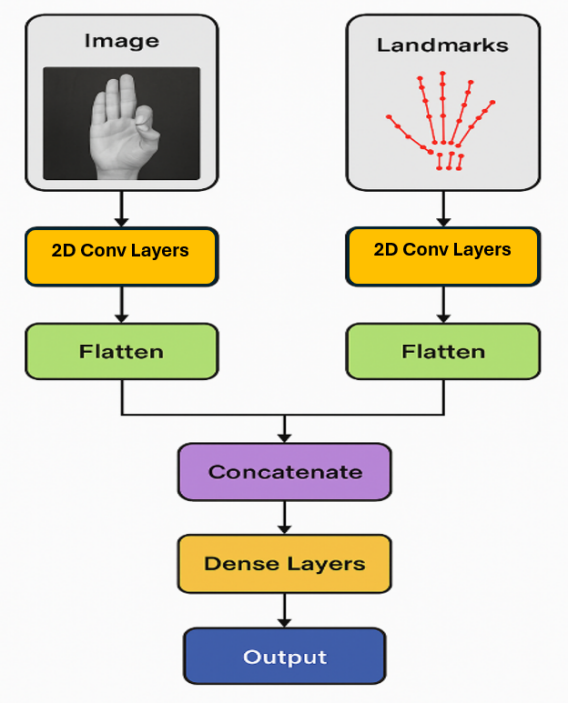

Figure 3. Flowchart of the proposed dual-input CNN architecture. The grayscale image and 3D landmarks are processed separately, concatenated, and classified.

5. Experiments and Results

5.1. Evaluation Metrics

To evaluate model performance, we used:

- **Accuracy:** Percentage of correctly predicted letters
- **Confusion Matrix:** To analyze per-class performance and common misclassifications
- **Training/Validation Curves:** To track model convergence and generalization

5.2. Baseline Model: Image-Only CNN

I first evaluated a simple CNN trained only on grayscale ASL images. While it performed decently on validation data, its accuracy degraded significantly in real-time usage due to overfitting to static images and limited spatial awareness.

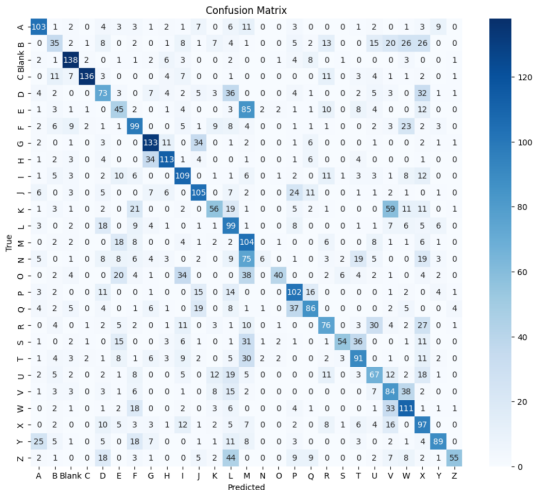

Figure 4. Confusion matrix for basic image-only CNN

5.3. Fine-tuned ResNet50 Model

I fine-tuned a ResNet50 model on the same dataset using colored ASL images. While ResNet50 achieved over 90% validation accuracy, its real-time performance was poor. The model was overly sensitive to background noise, hand orientation, and slight pose variations—it could not classify any gesture correctly. Additionally, it introduced significant inference latency, making it unsuitable for interactive applications.

This highlighted the limitations of transfer learning on sign language tasks where data diversity, speed, and spatial structure matter more than large model depth.

5.4. Dual-Input CNN (Main Model)

Using webcam or not, the dual-input CNN, which fuses grayscale image and hand landmark inputs, demonstrated the best performance among all tested models. It achieved stable predictions, high validation accuracy, and smooth real-time behavior even in uncontrolled settings.

Table 1. Classification Report for Dual-Input CNN Model

Class	Precision	Recall	F1-Score
A	0.94	0.99	0.96
B	0.93	0.95	0.94
C	0.99	0.99	0.99
D	0.87	0.91	0.89
E	0.93	0.94	0.94
F	0.91	0.95	0.93
G	0.95	0.95	0.95
H	0.95	0.95	0.95
I	0.96	0.94	0.95
J	0.91	0.97	0.94
K	0.93	0.94	0.94
L	0.95	0.96	0.96
M	0.85	0.92	0.88
N	0.84	0.93	0.87
O	0.94	0.90	0.92
P	0.90	0.98	0.94
Q	0.99	0.87	0.93
R	0.93	0.91	0.92
S	0.95	0.86	0.90
T	0.96	0.90	0.93
U	0.92	0.87	0.89
V	0.91	0.92	0.90
W	0.95	0.91	0.91
X	0.84	0.94	0.89
Y	0.97	0.91	0.94
Z	0.96	0.91	0.93
Accuracy		0.96	
Macro Avg	0.96	0.94	0.94
Weighted Avg	0.96	0.94	0.94

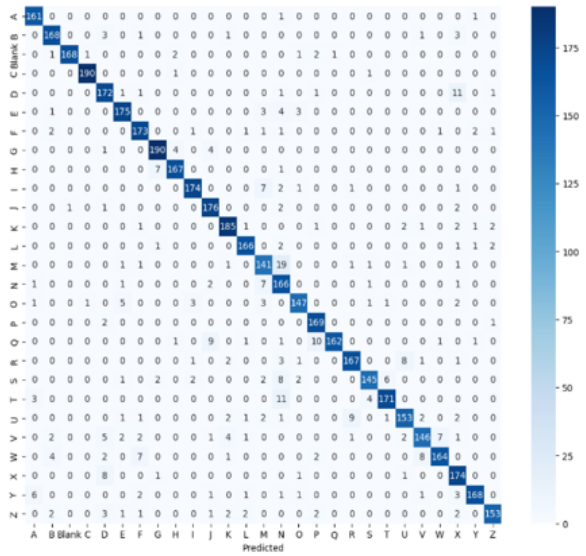

Figure 6. Confusion matrix for Dual-Input CNN

Figure 5. Classification Report for Dual-Input CNN Model

As seen in the classification report, the model achieved consistently strong performance across nearly all ASL classes. Notably, precision and recall remained high for both frequent and less common letters. These results were mirrored during real-time usage with webcam input, confirming the robustness of the model under noisy conditions.

5.5. Training Progress

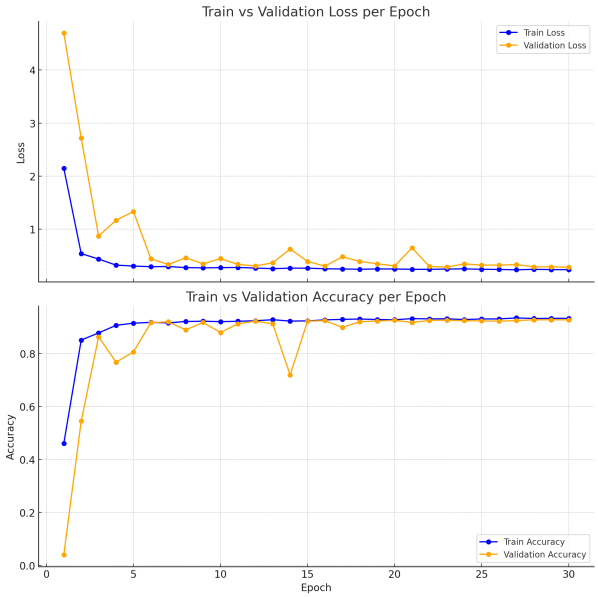

Figure 6. Training accuracy, validation accuracy, training loss, and validation loss curves for the Dual-Input CNN.

The training and validation curves show sharp convergence. The validation accuracy closely tracked training accuracy without signs of overfitting, which can be attributed to the augmentation strategy and regularization. The model reached over 95% validation accuracy around the 15th epoch before early stopping.

5.6. Real-Time Testing

Only the dual-input model generalized well to real webcam scenarios. Key observations:

- It maintained high prediction stability across different lighting conditions.
- The 1.5-second confirmation timer reduced false positives.
- The system provided responsive auditory feedback through text-to-speech.

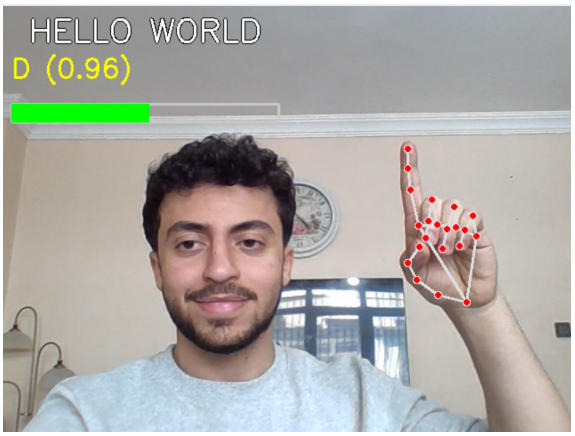

Figure 7. Live demonstration of model output during real-time webcam testing, showing sentence construction and voice feedback

These results demonstrate the practical viability of the dual-input architecture for real-time ASL recognition tasks.

6. Conclusion

In this project, we developed a real-time American Sign Language (ASL) recognition system using a novel dual-input Convolutional Neural Network (CNN) architecture. By fusing grayscale hand images with 3D landmark coordinates extracted via MediaPipe, our model was able to learn both visual texture and spatial joint structures, significantly enhancing its robustness in real-world scenarios.

I compared this dual-branch approach against multiple baselines, including a simple image-only CNN and a fine-tuned ResNet50 model. While these alternatives showed strong validation performance, they failed to generalize effectively in real-time webcam testing. In contrast, my proposed system demonstrated consistent, accurate predictions across different lighting conditions and hand orientations, supported by a stability-based confirmation mechanism and real-time speech feedback.

Our findings demonstrate the effectiveness of combining visual and spatial modalities in gesture recognition. The final system is efficient, responsive, and deployable in real-world contexts, offering practical accessibility improvements for the Deaf and hard-of-hearing communities.

Future work could extend this system to dynamic signs and full ASL sentences using video sequences and Recurrent Neural Networks (RNNs) or Transformers. Expanding the dataset with more user diversity and backgrounds will also further enhance robustness.

References

[1] P. C. Torres and C. A. Olvera, *American Sign Language Recognition with Deep Learning*, Computación y Sistemas, vol. 24, no. 3, pp. 1211–1220, 2020. https://www.scielo.org.mx/scielo.php?script=sci_arttext&pid=S1405-55462020000301211

[2] O. K. Oyedotun and A. Khashman, *Deep learning in vision-based static hand gesture recognition*, Expert Systems with Applications, vol. 122, pp. 112–134, 2018. https://www.sciencedirect.com/science/article/abs/pii/S0952197618301921

[3] R. Mittal and G. Balakrishnan, *Hand Gesture Recognition using MediaPipe and Deep Learning*, Journal of Artificial Intelligence and Data Science, vol. 2, no. 1, pp. 35–42, 2021. https://www.sciencedirect.com/science/article/pii/S2666990021000471

[4] F. Zhang et al., *MediaPipe Hands: On-device Real-time Hand Tracking*, Google Research Blog, 2020. https://google.github.io/mediapipe/solutions/hands

[5] K. He, X. Zhang, S. Ren, and J. Sun, *Deep Residual Learning for Image Recognition*, CVPR, 2016. https://arxiv.org/abs/1512.03385

[6] *pyttsx3 Text-to-Speech Library*. https://pypi.org/project/pyttsx3/

References