A real-time American Sign Language (ASL) alphabet recognition system for assistive communication using TinyML

Makhosazana Moyo[1*], *and Kago* Letlhaku[1]
[1]School of Mechanical, Industrial and Aeronautical Engineering, University of the Witwatersrand, South Africa

Abstract. Sign language is a critical communication tool for individuals with hearing impairments. This study focuses on developing a hand gesture recognition system to identify the American Sign Language (ASL) static alphabet using a Convolutional Neural Network (CNN) model deployed on a TinyML kit. The system is designed for real-time classification, making it a practical and accessible assistive technology. The model was tested at different distances, achieving an accuracy of 93.62% at 17.5 cm and 89.47% at 23 cm. Future work includes extending the system to recognise dynamic gestures and conducting user evaluations to improve robustness and accuracy.

1 Introduction

Communication is a fundamental human right. Yet over 1.5 billion individuals globally who live with hearing loss continue to face significant communication barriers in daily life [1]. Among the most empowering tools for the deaf and hard-of-hearing community is sign language, a rich visual language comprising hand gestures, facial expressions, and body movements. With more than 300 recognised sign languages worldwide [2], including widely used systems such as American Sign Language (ASL), sign languages play a pivotal role in promoting inclusion and self-expression among individuals with hearing impairments.

Hand Gesture Recognition (HGR) technology has emerged as a transformative innovation in this field, seeking to automate the interpretation of sign languages and enable seamless interaction between signers and non-signers. Early developments in HGR involved rule-based and vision-based approaches, but their limitations, especially in dynamic or uncontrolled environments, spurred the evolution toward machine learning-driven solutions [3, 4, 5]. Among these, Convolutional Neural Networks (CNNs) have become a dominant architecture, celebrated for their ability to extract spatial hierarchies and features essential for accurate classification of static and dynamic gestures [4, 6].

Despite the promise of HGR systems, several challenges persist. Vision-based models are sensitive to lighting variations, background noise, and camera resolution, which can degrade performance [5, 6]. Sensor-based systems like data gloves, though accurate, often

* Corresponding author: , Makhosazana.Moyo@wits.ac.za

compromise comfort, cost, and mobility [3], [4]. Furthermore, the accurate recognition of dynamic gestures, such as those used for letters “J” and “Z” in ASL, remains a computationally demanding task, particularly on low-power devices [5].

In response to these challenges, recent advancements in embedded AI and low-power microcontrollers such as the Tiny Machine Learning (TinyML) have enabled real-time sign language interpretation using low-cost, energy-efficient hardware. Common machine learning techniques such as Support Vector Machines (SVM), YOLOv5 and CNNs have been critical in the development of HGR systems [2, 4]. Unlike SVMs, which lack spatial feature extraction, CNNs have been proved to effectively identify intricate hand gesture patterns with better accuracy [2].

CNNs offer a lightweight and efficient solution for static ASL gesture recognition on low-cost microcontrollers. They are capable of direct classification with minimal computational overhead, making them ideal for deployment on TinyML platforms. In contrast, recent methods like YOLOv1-v5, while more powerful in detecting and localising objects in complex scenes, requires greater computational resources, which may exceed the capabilities of typical microcontrollers [7].

Recent studies [4, 8, 9, 10, 11] highlight the effectiveness of CNNs in ASL recognition. For example, Alsharif et al. [9] proposed a real-time ASL interpretation system combining YOLOv11 with MediaPipe hand keypoint tracking, achieving 98.2% accuracy with minimal latency. Aly and Fathi [10] introduced a hybrid CNN–Vision Transformer (ViT) architecture that reached 99.97% accuracy while maintaining high inference speed. Rheiner et al. [11] demonstrated that pose-guided MobileNetV3 models could achieve 99.98% accuracy on CPU alone, enabling real-time ASL detection without GPU acceleration. Sharma et al. [12] explored a TinyML-based wearable system that classified signs using microcontroller-compatible models, achieving 88% accuracy with ultra-low latency, highlighting the potential of TinyML in embedded assistive devices.

The potential applications of such systems are vast. From educational tools and assistive communication devices to integration into smart environments and public service kiosks, HGR systems can drastically enhance the accessibility of information and services for the deaf community [13]. More ambitiously, cross-lingual sign recognition and multilingual support could pave the way for global accessibility and cultural inclusion.

This paper presents a CNN-based ASL alphabet recognition system deployed on TinyML hardware, exploring its real-time capabilities, evaluating its robustness across varying conditions, and proposing avenues for future expansion to dynamic gesture recognition and support for other sign languages such as South African Sign Language (SASL).

2 Problem formulation

Real-time recognition of static sign language gestures using low-power embedded devices remains a challenge due to limited processing capacity, lighting variability, and gesture complexity. While traditional high-performance systems achieve high accuracy, their power demands and lack of portability hinder practical deployment in assistive applications. This research seeks to answer the question: *Can a CNN deployed on a TinyML platform accurately recognise static ASL gestures in real-time using low-cost hardware?*

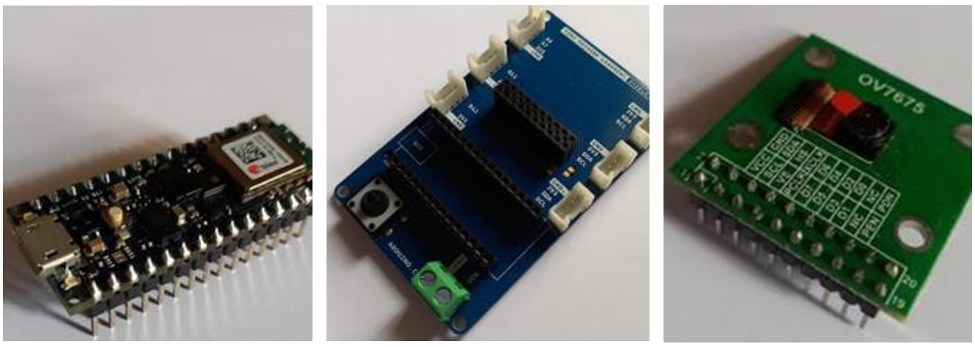
The study specifically addresses the need for a lightweight, cost-effective system that maintains high classification accuracy under constrained computational resources. The focus is on static ASL alphabets, while dynamic gestures are reserved for future work due to hardware limitations.

3 Methodology

The proposed system for real-time static ASL alphabet recognition was developed in five main phases: hardware setup, data acquisition, image pre-processing, model training, and model deployment and testing. The system was implemented on an Arduino Nano 33 BLE Sense Lite, which served as the primary processing unit for deploying the TinyML model. To capture hand gesture images, an OV7675 camera module was connected to the board through a TinyML shield. The trained CNN model was optimised for execution on this low-power hardware. This methodology guaranteed compatibility with resource-constrained microcontrollers while maintaining classification accuracy.

3.1 Hardware setup

The hardware platform comprises of an Arduino Nano 33 BLE Sense Lite microcontroller, equipped with built-in sensors, and paired with a TinyML shield to interface an OV7675 camera module. This setup facilitated real-time image capture of hand gestures in grayscale format. Figure 1 shows the Arduino Nano microcontroller, TinyML shield and the OV7675 camera module.



(a) (b) (c)

Fig. 1. Images of the (a) Arduino Nano 33 BLE Sense Lite microcontroller, (b) TinyML shield, and (c) OV7675 camera module [17].

3.2 Data collection

A custom dataset of 24 static ASL alphabet gestures, excluding dynamic letters “J” and “Z”, was collected using the OV7675 camera, resulting in 24 gesture classes. Each gesture class contained 20 grayscale images captured at two different distances, 17.5 cm and 23 cm, to evaluate the model's distance robustness.

3.3 Image pre-processing

Captured images were resized from 320×240 to 160×120 pixels and converted to grayscale to reduce computational overhead and optimise model performance on the embedded device. Pre-processing was done using code uploaded to the Arduino Nano, which managed real-time data handling and image formatting. Figure 2 shows the raw image and the pre-processed image which is used as an input for the dataset.

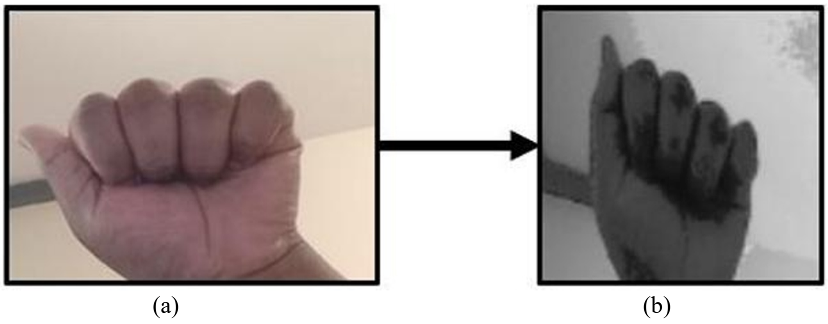


Fig. 2. (a) Raw image, (b) pre-processed image in grayscale.

3.4 Model training

Model training was conducted using Google's Teachable Machine — a web-based platform designed for rapid prototyping of machine learning models. The system employs a lightweight, MobileNet-based CNN architecture, well-suited for real-time image classification on resource-constrained devices. The model was trained on custom ASL alphabet data with the following configuration:

- 300 epochs
- Batch size of 16
- Learning rate of 0.001
- ReLU activation in convolution layers and Softmax in the output layer

Upon completion, the model was exported as a quantised TensorFlow Lite (TFLite) format optimised for deployment on the Arduino Nano 33 BLE Sense microcontroller. The integration between Teachable Machine and the Arduino IDE enabled seamless dataset creation and real-time image capture from the OV7675 camera module. A sample of the dataset used for training is shown in Figure 3.

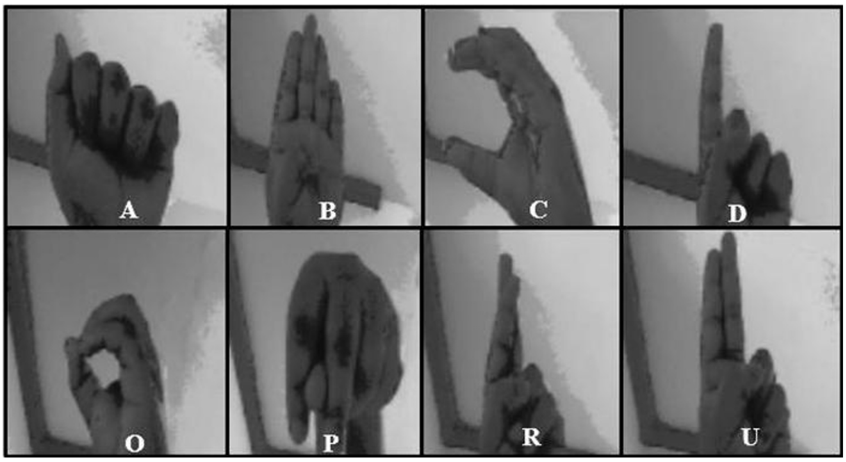


Fig. 3. An example of the dataset which is used to train the model.

3.4.1 CNN architecture

A CNN architecture, adapted from an existing framework which detects books and cups, was employed to identify the corresponding ASL alphabet for the input hand gesture [14]. The CNN architecture for hand gesture alphabet recognition system is shown in Figure 4. Initially, the model extracts specific features from the input image, which are then compared against the dataset used to train the model. Through multiple layers of processing, the model outputs the matching class for the input gesture, thereby predicting the letter represented by the hand gesture in ASL.

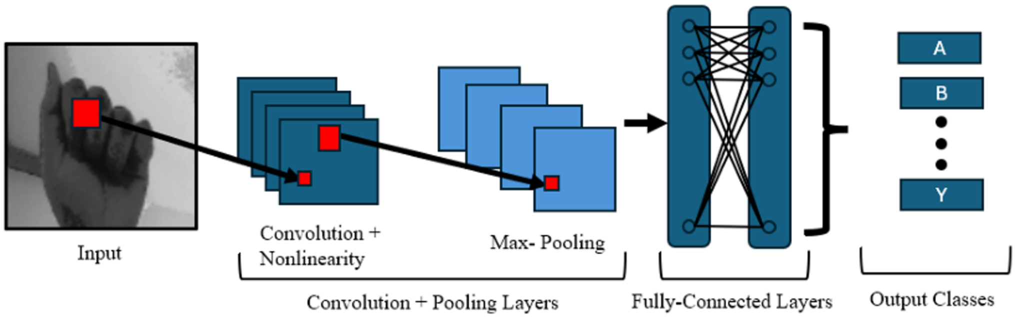


Fig. 4. CNN architecture for hand gesture alphabet recognition system.

3.4.2 Model deployment and real-time testing

During real-time testing, the trained TFLite model was deployed on the Arduino Nano using the Arduino IDE. The embedded model captured live input from the camera, classified it, and displayed the corresponding ASL letter via a serial monitor. Performance of the model was tested at both 17.5 cm and 23 cm from the camera to assess robustness to distance. The pseudocode in Algorithm 1 shows the flow of real-time testing.

Algorithm 1: Real-Time testing

```
1:  Deploy trained model: Deploy trained model and embed it onto the microcontroller using
    Arduino IDE
2:  Initialise: The OV7675 camera module
3:  For every image captured by the camera do
4:    Extract feature: Extract features on the captured imaged to be used for recognition
5:      While classes of alphabets are not complete and match not found do
6:          Use CNN to identify which class matches the extracted feature
7:      end while
8:    Display the output
9:  End for
10: Convert the output to percentage
```

3.5 Validation and evaluation

Validation was done using both visual inspection and numerical accuracy derived from classification outputs. Accuracy was calculated per class and across the dataset using the equation:

$$\text{Accuracy (\%)} = \frac{A + K}{256} \times 100 \qquad (1)$$

Where A is the observed model output value ranging from -128 to 128, and K is a bias constant equal to 128. The value 256 represents the normalisation range (−128 to +128), and the formula converts the output to a percentage accuracy score.

Results confirmed a mean accuracy of 93.62% at 17.5 cm and 89.47% at 23 cm, affirming the system's capability for accurate gesture classification under constrained computational resources.

4 Results and discussion

The developed ASL hand gesture recognition system was evaluated for real-time performance on a TinyML platform using a custom dataset of 24 static ASL alphabet gestures. The CNN model trained via Teachable Machine demonstrated strong classification performance when deployed on the Arduino Nano 33 BLE Sense Lite microcontroller. This section delves into model validation, real-time performance, distance sensitivity as well as per-class accuracy.

4.1 Validation accuracy

During model training, validation accuracy peaked at 95%, while the real-time test model achieved 93.62% accuracy at 17.5 cm, the optimal distance for the OV7675 camera. These results confirm that the CNN effectively extracted spatial features from pre-processed grayscale images and classified them accurately in real time.

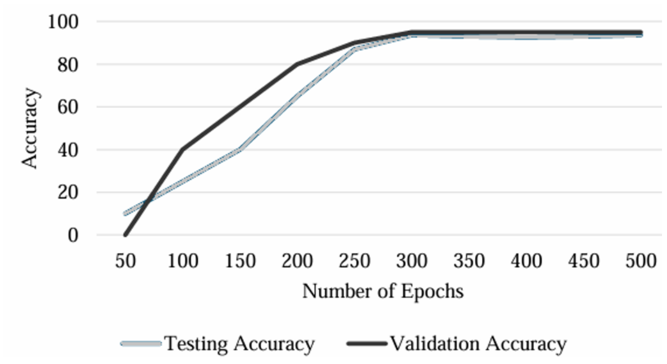


Fig. 5. Percentage accuracy per epochs.

The model's learning curve, shown in Figure 5, demonstrates a stable convergence of validation and testing accuracies. The high validation accuracy suggests that the CNN was well-optimised for the dataset, with no indication of overfitting despite limited sample sizes and constrained hardware resources.

To evaluate model robustness, a comparative analysis was conducted against alternative classifiers commonly used in gesture recognition, namely SVM, Traditional Neural Network

(NN), and Hidden Markov model (HMM). Table 1 presents a comparison with results reported in other studies. SVMs reached 92% accuracy in prior studies but are limited by their inability to extract spatial hierarchies, which are crucial in recognising nuanced hand gestures [16]. Traditional NNs achieved 90.6% accuracy, but lacked convolutional layers, hindering their ability to process image-based features effectively [15]. HMMs which are well-suited for recognising dynamic gestures in sign language, as they capture temporal patterns and transitions between sequential hand movements had a 90% and greater accuracy in specific ISL (Indian Sign Language) recognition tasks [5]. However, HMMs rely heavily on handcrafted features and struggle to capture spatial information in images, whereas CNNs automatically learn and extract complex spatial patterns directly from raw image data, making them more effective for visual tasks like hand gesture recognition [5].

By contrast, the proposed CNN model autonomously extracts spatial features and achieves a strong balance between accuracy, computational efficiency, and real-time performance on microcontrollers, with 93.62% accuracy under ideal conditions.

Table 1. Method comparison using percentage accuracy.

Classification Method	Test Accuracy (%)	Comments
Support Vector Machine (SVM) [16]	92.0	Lacks spatial feature learning
Traditional Neural Network [15]	90.6	No convolutional layers
Hidden Markov Model [5]	90.0	Rely heavily on handcrafted features
Proposed CNN Model	93.62	Balanced accuracy and efficiency

These findings reaffirm the selection of CNNs for the TinyML-based hand gesture recognition, especially for multi-class static gesture classification. Their ability to operate under resource constraints while preserving high accuracy makes them well-suited for real-world, low-power assistive applications.

4.2 Real-time performance and distance sensitivity

The model was tested at two focal distances, 17.5 cm and 23 cm, to assess robustness under different input proximities. Results showed a drop in accuracy from 93.62% to 89.47% as the distance increased, indicating a negative correlation between gesture distance and recognition performance.

At 17.5 cm, the system achieved its highest classification accuracy of 93.62%, attributed to the optimal resolution and feature visibility captured by the camera at this proximity. When the distance increased to 23 cm, accuracy dropped to 89.47%, reflecting the effect of diminished feature resolution and clarity in image capture. The relationship between distance and accuracy is visually summarised in Figure 6. Table 2 shows the corresponding class and alphabet letter.

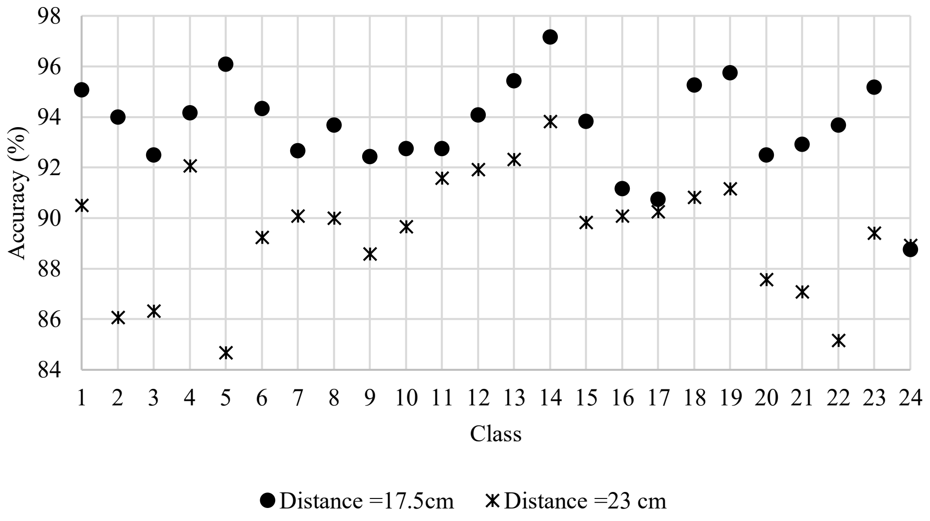


Fig. 6. Accuracy at 17.5 cm and 23 cm.

Table 2. Alphabet letter and the corresponding class.

Letter	A	B	C	D	E	F	G	H	I	K	L	M
Class	1	2	3	4	5	6	7	8	9	10	11	12
Letter	N	O	P	Q	R	S	T	U	V	W	X	Y
Class	13	14	15	16	17	18	19	20	21	22	23	24

This degradation in performance with increased distance can be explained by reduced image scale and feature density. As the hand moves further from the camera, the effective size of the hand in the image frame decreases, leading to fewer distinguishable features being captured. Given the image is pre-processed to a low resolution (160×120 pixels) for TinyML compatibility, further distance reduces the model's ability to discern subtle differences in gestures, especially between similar shapes (e.g., "K" and "V"). Additionally, limitations of the OV7675 camera module results in reduced accuracy. The OV7675 camera has a fixed focus range and operates at a low resolution (640×480, down-sampled in pre-processing). Its limited field of view and basic optics are not ideal for variable-distance object detection, particularly in low-light or cluttered backgrounds [17].

Despite these limitations, the system maintains a relatively high degree of robustness. Over 70% of the ASL alphabet classes maintained over 90% accuracy at 23 cm, suggesting the CNN model generalises reasonably well even with slight degradation in input quality.

These findings emphasise the importance of distance calibration and dataset diversity during model training. Incorporating a wider range of hand sizes, angles, and lighting conditions could further enhance the model's real-world applicability. Moreover, integrating adaptive pre-processing or autofocus hardware could compensate for the loss in resolution at increased distances.

4.3 Per-class accuracy

An analysis of the system's performance across individual ASL alphabet classes reveals important insights into how gesture complexity and visual similarity affect recognition accuracy. As shown in Table 3, the system demonstrated consistently high accuracy across most gesture classes when tested at the optimal 17.5 cm distance, with values ranging between 88.75% and 97.17%. However, some classes exhibited significantly better performance than others, reflecting variations in hand geometry, gesture clarity, and inter-class similarity.

Table 3. Per-class testing accuracy at 17.5 cm and 23 cm away from camera module.

		Testing Accuracy (%)	
Class	Alphabet	17.5 cm	23.0 cm
1	A	95.08	90.50
2	B	94.00	86.08
3	C	92.50	86.33
4	D	94.17	92.08
5	E	96.08	84.67
6	F	94.33	89.25
7	G	92.67	90.08
8	H	93.67	90.00
9	I	92.42	88.58
10	K	92.75	89.67
11	L	92.75	91.58
12	M	94.08	91.92
13	N	95.42	92.33
14	O	97.17	93.83
15	P	93.83	89.83
16	Q	91.17	90.08
17	R	90.75	90.25
18	S	95.25	90.83
19	T	95.75	91.17
20	U	92.50	87.58
21	V	92.92	87.08
22	W	93.67	85.17
23	X	95.17	89.42
24	Y	88.75	88.92
Average Accuracy		93.62	89.47

Certain letters such as “O” (97.17%), “S” (95.25%), and “T” (95.75%) consistently achieved top accuracy scores. These gestures possess distinct, easily identifiable shapes with minimal variation in user presentation. For example, the ASL sign for “O” forms a circular shape with the fingers, creating a visually unique closed loop that the CNN could reliably detect and distinguish from other letters, as shown in Figure 7.

Similarly, the gestures for “S” (fist) and “T” (thumb tucked under index finger) are geometrically distinct and occupy compact, non-ambiguous spatial configurations, as shown in Figure 7. These characteristics reduce the likelihood of feature overlap with other classes, improving classification reliability.

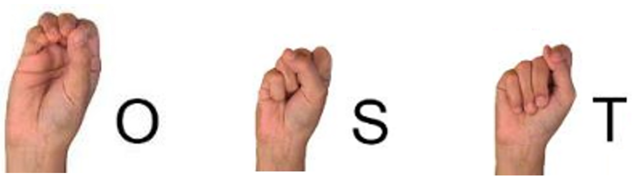


Fig. 7. Hand gestures for “O”, “S” and “T”, illustration adapted from [18].

On the lower end of the accuracy spectrum, letters like “K” (92.75%), “V” (92.92%), and “W” (93.67%), shown in Figure 8, exhibited slightly reduced accuracy, especially at increased distances. These gestures share similar finger orientations, often involving two or three extended fingers in a V or W shape, with minor differences in thumb placement or finger angle. When captured using a low-resolution grayscale image (160×120), these distinctions become less apparent to the model, leading to frequent misclassifications.

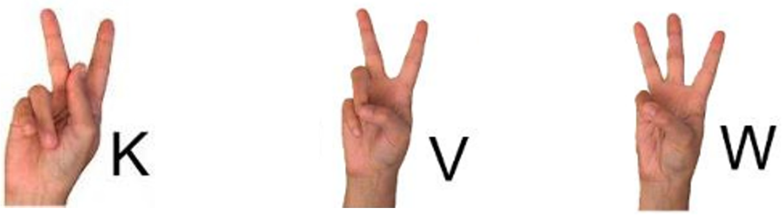


Fig. 8. Geometric similarity between “K”, “V” and “W”, illustration adapted from [18].

These challenges reflect common issues in vision-based gesture classification, where fine-grained inter-class variation must be learned by the model under limited input resolution. In the case of “K” and “V,” both shapes can appear nearly identical depending on the angle of the hand, lighting conditions, or partial obstruction in the camera frame.

The impact of hand-to-camera distance also varied by class. While many letters retained high accuracy (>90%) at 23 cm, more complex shapes like “W” and “E” showed sharper declines to 85.17% and 84.67%, respectively. This further emphasises the dependence of certain gestures on close-range image clarity and the limitations of the OV7675 camera's fixed-focus lens in capturing fine hand articulation at longer distances [12].

These per-class performance trends suggest that while the CNN architecture generalises well, its effectiveness is inherently tied to the distinctiveness of the input gesture and the visual separability of classes in the dataset. The consistent performance across 24 classes validates the robustness of the trained model; however, future improvements could involve:

- augmenting training data with additional angles and hand orientations,
- including pose-invariant features or applying attention mechanisms within the model architecture, and

– leveraging multi-frame temporal modelling for difficult gestures, even among static classes, to capture subtle motion or positioning cues.

These findings reaffirm the selection of CNNs for TinyML-based hand gesture recognition, especially for multi-class static gesture classification. Their ability to operate under resource constraints while preserving high accuracy makes them well-suited for real-world, low-power assistive applications.

5 Conclusions and recommendations

This work addressed the challenge of performing real-time gesture recognition on low-power hardware by deploying a CNN on TinyML. The system achieved a promising balance between accuracy, efficiency, and portability, making it a viable candidate for assistive communication technologies in resource-constrained environments. However, dynamic gesture recognition and adaptation to variable lighting conditions remain open challenges for future work.

The model achieved a validation accuracy of 95% and real-time testing accuracy of 93.62% at the optimal focal distance of 17.5 cm, confirming its effectiveness in classifying 24 static ASL alphabet gestures. This work determined that the accuracy declined to 89.47% at 23 cm, highlighting the system's sensitivity to hand-to-camera distance and the limitations of the OV7675 low-resolution camera module.

Certain gestures such as "O," "T," and "S" were classified with high confidence due to their distinct geometries, while visually similar signs like "K" and "V" presented greater classification challenges. A comparative analysis also demonstrated that the proposed CNN outperforms traditional classifiers like SVM and standard neural networks, particularly in handling spatial features and multi-class classification.

To improve the robustness and applicability of the system in real-world conditions, the following enhancements are recommended:

– Extend the system to include dynamic gestures such as "J" and "Z", possibly through integration of temporal models like Long Short-Term Memory (LSTM) networks or 3D CNNs.
– Expand the dataset to incorporate variations in hand size, skin tone, lighting conditions, and background complexity to improve generalisability.
– Replace the OV7675 camera module with higher-resolution or auto-focus cameras to better capture fine gesture details, especially at longer distances.
– Investigate extending recognition to support SASL, which uses two-handed gestures and presents increased linguistic complexity.
– Conduct usability testing with individuals from the deaf community to evaluate the system's real-world effectiveness, accessibility, and user satisfaction.

The authors would like to express sincere gratitude to the School of Mechanical, Industrial and Aeronautical Engineering Research Committee at the University of the Witwatersrand for their financial support in funding the equipment used in this research. Their contribution was instrumental in the successful development and implementation of the hand gesture recognition system.

References

1. W.H.O, Deafness, https://www.who.int/news-room/facts-in-pictures/detail/deafness (2024)

2. Sign Solutions, What are the different types of sign language?, https://www.signsolutions.uk.com/what-are-the-different-types-of-sign-language/ (2024)
3. B. Shi, X. Chen, Z. He, R. Han, Development of magnetic-sensor-based hand gesture recognition system for sign language, in Proceedings of the IEEE International Electrical and Energy Conference. (CIEEC), Hefei, China, 12-14 May (2023), **2302–2305**.
4. M. Bansal, S. Gupta, Detection and recognition of hand gestures for Indian Sign Language recognition system, in Proceedings of the IEEE International Conference on Signal Processing, Computing and Control (ISPCC), Solan, India, 07-09 October (2021)**, 136–140**.
5. K. Tripathi, N. Baranwal, G.C. Nandi, Continuous dynamic Indian Sign Language gesture recognition with invariant backgrounds, in Proceedings of the International Conference on Advances in Computing, Communications and Informatics (ICACCI), Kochi, India, 10-13 August (2015), **2211–2216**.
6. P.S. Neethu, R. Suguna, D. Sathish, An efficient method for human hand gesture detection and recognition using deep learning convolutional neural networks. Soft Comput. **24**, 15239–15248 (2020). https://doi.org/10.1007/s00500-024-10038-0
7. H.D. Alon, M.A. Ligayo, M.P. Melegrito, C.F. Cunanan, E.E. Uy II, Deep-hand: A deep inference vision approach of recognising a hand sign language using American alphabet, in Proceedings of the IEEE International Conference on Computational Intelligence and Knowledge Economy (ICCIKE), Dubai, United Arab Emirates, 17-18 March (2021), **373–377**.
8. A. Jain, A. Sethi, D.K. Vishwakarma, A. Jain, Ensembled neural network for static hand gesture recognition, in Proceedings of the IEEE in Proceedings of the 12th International Conference on Computing Communication and Networking Technologies (ICCCNT), Kharagpur, India**,** 06-08 July (2021), **1–5**.
9. B. Alsharif, L. Peng, M. Mohaisen, F. Alabdulmohsin, Real-time American Sign Language interpretation using deep learning and key-point tracking. *Sensors* **25**, 2138 (2025). https://doi.org/10.3390/s25072138
10. M. Aly, I.S. Fathi, Recognising American Sign Language gestures efficiently and accurately using a hybrid transformer model. *Sci. Rep.* **15**, 20253 (2025).
11. J. Rheiner, T. Dietz, A. Kroll, From pixels to letters: A high-accuracy CPU-real-time American Sign Language detection pipeline. *Mach. Learn. Appl.* **20**, 100650 (2025). https://doi.org/10.1016/j.mlwa.2025.100650
12. S. Sharma, R. Gupta, A. Kumar, A TinyML solution for an IoT-based communication device for hearing impaired. *Expert Syst. Appl.* **246**, 123147 (2024). https://doi.org/10.1016/j.eswa.2024.123147
13. P. Muralidhar, A. Saha, P. Sateesh, Customisable dynamic hand gesture recognition system for motor impaired people using siamese neural network, in Proceedings of the IEEE International Conference of Artificial Intelligence and Information Technology (ICAIIT), Yogyakarta, Indonesia**,** 13-15 March (2019), **354–358**.
14. *S. Adesola,* TinyML on Arduino Nano 33 BLE Sense with Teachable Machine*,* https://github.com/adesolasamuel/TinyML-on-Arduino-Nano-33-BLE-Sense-with-Teachable-Machine (2024)
15. Z. Wang, B. Chen, J. Wu, Effective Inertial Hand Gesture Recognition Using Particle Filtering Based Trajectory Matching. J. Electr. Comput. Eng. **2018**, 6296013 (2018). https://doi.org/10.1155/2018/6296013

16. M.H. Rahman, J. Afrin, Hand Gesture Recognition using Multiclass Support Vector Machine. Int. J. Comput. Appl. **74**, 39–43 (2013). https://doi.org/10.5120/12852-9367
17. Arducam, *OV7675 20-pin DVP Camera Module for Arduino*, https://store.arduino.cc/products/arducam-camera-module (2024)
18. Start ASL, *Sign Language Alphabet | 6 Free Downloads to Learn it Fast*, https://www.startasl.com/american-sign-language-alphabet/ (2024)