A comparison of common classifier models for hand pose estimation for the New Zealand Sign Language alphabet

Ryan B right1\*

1 Department of Mechanical Engineering, University of Canterbury, Christchurch, New Zealand

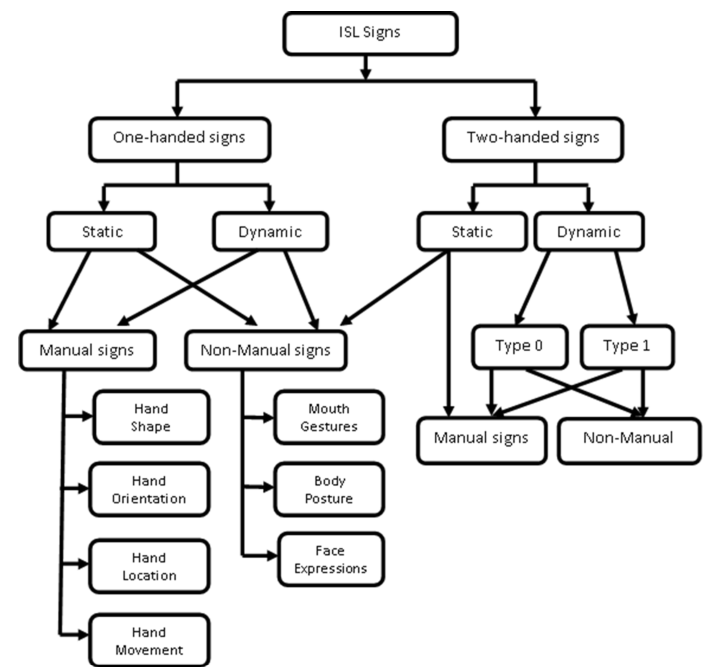
\* Corresponding Author Email: rbr115@uclive.ac.nz

|  |
| --- |
| *ABSTRACT:* |

KEYWORDS: New Zealand Sign Language; MediaPipe Hands; Hand Pose Estimation

# Introduction

New Zealand Sign Language (NZSL) is one of New Zealand’s three official languages, yet digital tools that provide accessibility to the language remain limited. There is a significant language barrier between people who rely on NZSL and those who don’t. Unlike spoken language, sign languages pose a unique set of challenges for translation tools. As shown in Figure 1, NZSL relies on precise hand positioning, movement, facial shapes, and body poses to communicate [1][2]. Additionally, dictionaries are challenging to use and develop due to NZSL not being a written language. This makes consistent, real-time classification a non-trivial task.



NZSL

Figure 1. Classification of sign language word forms. Adapted from [1]

Previous efforts for other sign languages like American Sign Language (ASL) and Chinese Sign Language (CSL) show a wide range of approaches and methodologies. For example, the use of Hough transformations and CNNs [3], the use of 3D hand tracking gloves [4], and the use of various neural network models [5]. Notably, camera-based methods in papers reviewed in [1], all focus on single hand signs. This poses a challenge for NZSL which has a high amount of two handed signs compared to more mainstream sign languages like ASL.

Presented is a complete pipeline including data gathering, training and evaluation with additional scripts for evaluating training data balance and correcting camera lens distortion. The current method detects one or two handed static manual signs. This is done using MediaPipe to generate hand landmarks, and then a multilayer perceptron (MLP) network to classify the hand pose. The method is evaluated on validation data and real time webcam input, showing high accuracy in controlled environments and highlighting some key limitations.

# Methodology

## Environment

Table 1 lists the development environment used. Only the key python modules are specified for simplicity. The full project can be found on GitHub [6].

Table 1. Environment used in pipeline development

|  |  |
| --- | --- |
| Component | Description |
| Operating System | Windows 11 |
| Web Camera model | Built-in 1080p webcam |
| Computer model | Toshiba Portege X30 E |
| CPU | Intel i5-8250U |
| Python Ver | 3.12.9 |
| TensorFlow Ver | 2.19 |
| OpenCV-Python Ver | 4.11 |
| MediaPipe Ver | 0.10.21 |

## Inference pipeline and architecture

The inference pipeline combines MediaPipe and a trained MLP classifier to process webcam input and detect hand poses in real time. Key steps in the pipeline are illustrated in Figure 2.

Figure 2. Flow of presented inference pipeline

The webcam video feed is loaded using the python library OpenCV-python. Then the lens distortion is corrected for using OpenCVs undistort() function and a precalibrated distortion matrix. The distortion matrix is calculated from a reference chess board pattern and measuring deviations from straight lines.

Google's MediaPipe Hands library [7] was selected for extracting landmark data. It provided a lightweight, real-time hand-tracking solution capable of detecting 21 landmarks per hand, Figure 3, with relatively high accuracy. Compared to training a custom model from scratch, MediaPipe reduced the problem from the domain of camera pixels to that of hand landmark coordinates.

A hand with dots and lines

AI-generated content may be incorrect.

Figure 3. Indexed Landmarks from MediaPipe Hand’s model.

To ensure the proposed method is flexible and able to deal with different hand locations and sizes, a normalization algorithm is employed. Normalization begins by translating the coordinate system to the midpoint between the wrists, removing positional variance in the data set. Next, the maximum Euclidean norm from the origin to any landmark is used to scale the landmarks, achieving scale invariance while preserving relative geometry, Figure 4.

A comparison of a graph

AI-generated content may be incorrect.

Figure 4. Side by side comparison of x-y projection of original and normalized landmarks. Right hand drawn with skeleton.

One important edge case to cover in normalization is single hand signs. When there is only one hand detected, its landmarks get stored in place for both hands. This allows the normalization algorithm to function using the same rules for both single and double hands.

Lastly a neural network is employed to take the landmarks and classify the hand pose. Due to the structured, low-dimensional input and lack of temporal sequence data, an MLP neural network was selected

The implemented network has an input layer of 126 nodes, storing x,y,z coordinate data for all 21 landmarks for both hands. The hidden layers are outlined below.

* 256 node dense layer with ReLU activation
* 30% dropout layer
* 128 node dense layer with ReLU activation
* 30% dropout layer
* Output layer with Softmax activation

The output layer applies the Softmax activation to produce a class probability distribution.

## Data collection and processing

Due to an absence of publicly available datasets specific to NZSL, a custom data collection tool was developed. This tool captures hand pose data and saves the normalized landmarks from each detected frame as individual .json files.

A total of 4,500 labeled samples were collected, each containing normalized coordinates of 21 landmarks per hand, along with a label and timestamp. The data was stored in json files for ease of loading and storage of additional metadata. The data set includes labels for the alphabet letter signs: A,E,I,O,C,M, and N. Additionally there is a label for ‘no pose’ which was saved under ‘Z’ and is displayed as Null Pose. This selection of classes includes a mix of single hand signs, overlapping hand signs, and highly similar hand shape signs. These letters were chosen to best evaluate the models flexibility and precision in distinguishing hand shapes.

To mitigate class imbalance, which can bias the model, an approximately uniform distribution across all classes was targeted during data gathering. The final class distribution is shown in   
Figure 5.

A graph of blue bars

AI-generated content may be incorrect.

Figure 5. Frequency histogram of labeled landmark hand poses.

To ensure low false positive rates on the trained classes from the model, a large amount of null pose reference samples was needed. These samples were critical for distinguishing between intentional sign poses and idle hand positions. The null pose training data causes denser feature maps giving rise to better class boundary separation, as illustrated by Figure 6.

A diagram of different colored dots

AI-generated content may be incorrect.

Figure 6. Illustration of two feature maps showing three classes’ labeled data (Orange, Blue, and Grey) and possible class boundaries between them (Black).

## Training

Of the full dataset, 80% of the samples were randomly allocated for training and 20% for validation. To increase the effective size and diversity of the training data, data augmentation techniques were applied. Each training sample was duplicated multiple times with slight variations, increasing the size of the training set by 300%. Augmentations included random translations to the left- and right-hand landmarks, followed by the addition of Gaussian noise to the (x,y,z) coordinates of each landmark. These transformations preserved the underlying hand pose while simulating natural variation and sensor noise, thereby improving model robustness.

The model was trained using the Adam optimizer, a sparse categorical cross-entropy loss function (1), a batch size of 32, and the model was trained over 10 epochs. Training was conducted using TensorFlow/ Keras, and key metrics such as training loss, validation loss, and accuracy were tracked across all epochs.

*For is the softmaxed model prediction vector, N is the number of classes*

Early stopping and learning rate scheduling were considered during model development but ultimately not applied, as the model converged reliably within a small number of epochs without signs of instability.

# Results

Running the inference script shows the system consistently achieves high confidence when detecting hand poses, Figure 7. The system correctly detects all seven trained classes over a wide range of hand pose variations.

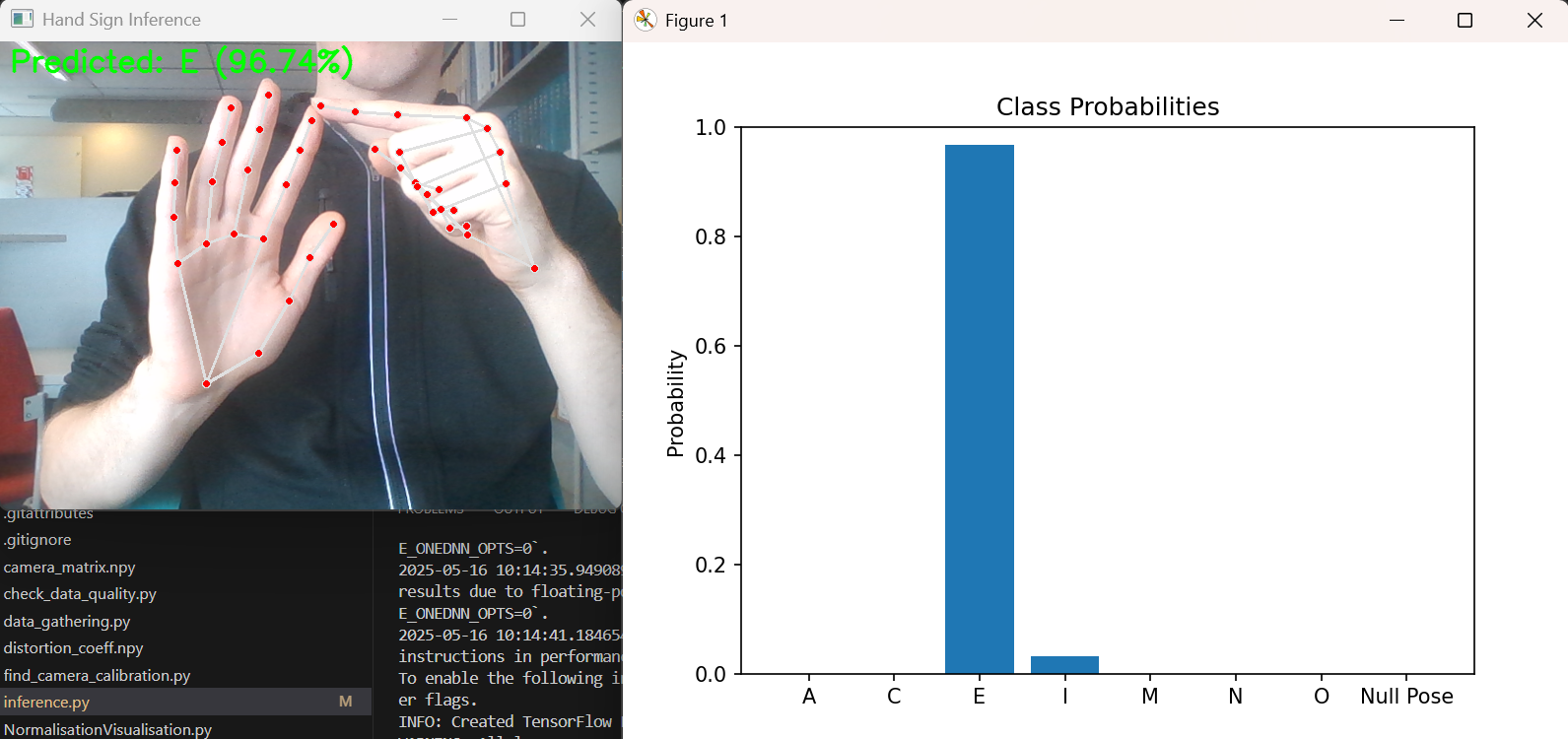


Figure 7. Live demonstration of the proposed inference pipeline. The model shown was trained with the standard hyperparameters outlined in the methodology.

The model shows high confidence in class identification, exemplified by the training history, Figure 8. The training history stores key metrics for tracking training performance. After ten epochs, the training is largely finished with high accuracy and low loss.

A graph of a graph

AI-generated content may be incorrect.

Figure 8. Accuracy and loss curves for both training and validation sets. Training was conducted using the baseline hyperparameters of 10 epochs and 300% data augmentation.

Overfitting occurs when the model begins to capture noise or high-frequency signals that are specific to the training data at the expense of generalization [8]. This behavior is typically observed when the validation loss begins to increase while training loss continues to decrease. In Figure 9, this pattern is visible beyond 12 epochs, indicating that the model becomes overfitted and more sensitive to minor perturbations in landmark locations, degrading real-time inference accuracy.

A graph with blue and orange lines

AI-generated content may be incorrect.

Figure 9. Loss curves for a model trained over 120 epochs with 300% data augmentation. Overfitting is observed beyond epoch 12.

To identify good training hyperparameters, models were trained with varying degrees of data augmentation. Figure 10 shows the relationship between augmentation factor and the epoch at which overfitting begins. An augmentation factor of 3 with 10 epochs was found to produce a model with minimal overfitting behavior.

A graph with blue dots

AI-generated content may be incorrect.

Figure 10. The relationship between data augmentation factor and onset of overfitting. Points represent the epoch where validation loss exceeds training loss.

Using the standard training hyperparameters, the best model performance was able to achieve 99.25% accuracy on the validation data with a loss of 0.0396 across 900 samples. Live inference showed some shortcomings of the model. The model would often miss identify the hand pose for “O” as “I”,   
Figure 11, showing high sensitivity to palm orientation.

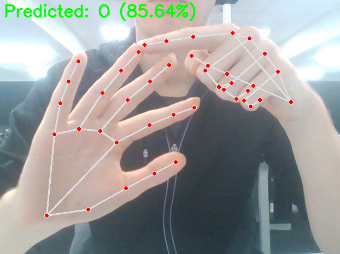
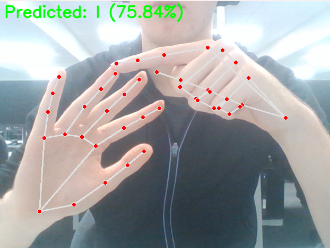


Figure 11. (Left) Miss identified “O” hand shape as “I”. (Right) Correctly identified hand shape “O” as “O”

Testing across different people showed similar live performance with slightly different hand proportions reducing the stability of “I” and “O” classes again. Notably “A”, “E”, “C”, “M”, and “N” were very consistent, Figure 12, across different lighting conditions, people, and backgrounds.

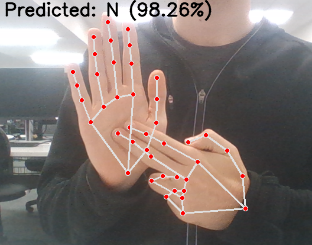
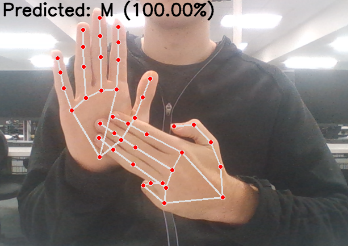


Figure 12. (Left) Correctly classified “M” hand pose. (Right) Correctly classified “N” hand pose

One key limitation of the system is its usage of MediaPipe. Landmark detection degrades significantly when hands are occluded or in contact, which is a known failure mode of the MediaPipe Hands model. As illustrated in Figure 13, landmark accuracy suffers during poses with self-occlusion or contact, significantly limiting detectable signs.



Figure 13. MediaPipe loosing track of hand in high contact and occlusion hand pose.

# Future Development

To further the proposed system, some further development is suggested based on the presented limitations and performance covered.

An alternative Landmark detector could be employed, replacing Mediapipe with a more flexible detector. [9]

Different classifier architectures could be employed to leverage their strengths. Potential architectures include K Nearest Neighbors and Support Vector Machines.

Preprocessing can be done to either remove the background and other noise, or to detect additional features like facial expression and hand-body relative pose to better reflect the multimodal nature of NZSL as outlined in Figure 1.

Lastly, an exploration into better representations of hand poses would be valuable. For example, disregarding the z channel of landmark data, a joint linkage angle representation of hand poses [10], or a temporal model of the hands motion [11].

# Conclusion

# References

[1] Wadhawan A, Kumar P (2021) Sign Language Recognition Systems: A Decade Systematic Literature Review. *Arch Computat Methods Eng* **28**, 785–813.

[2] Cheok, M., Omar, Z., & Jaward, M. (2017). A review of hand gesture and sign language recognition techniques. International Journal of Machine Learning and Cybernetics, 1–23.

[3] Munib Q, Habeeb M, Takruri B, Al-Malik HA (2007) American sign language (ASL) recognition based on Hough transform and neural networks. Expert Syst Appl 32(1):24–37

[4] Oz C, Leu MC (2007) Linguistic properties based on American Sign Language isolated word recognition with artificial neural networks using a sensory glove and motion tracker. Neurocomputing 70(16):2891–2901

[5] Wu J, Tian Z, Sun L, Estevez L, Jafari R (2015) Real-time American Sign Language recognition using wrist-worn motion and surface EMG sensors. In: IEEE 12th international conference on wearable and implantable body sensor networks (BSN), pp 1–6

[6] MY GITHUB REPO

[7] Zhang F, Bazarevsky V, Vakunov A, Tkachenka A, Sung G, Chang C-L, & Grundmann M (2020). MediaPipe Hands: On-device real-time hand tracking. In *arXiv [cs.CV]*. http://arxiv.org/abs/2006.10214

[8] Ian G, Yoshua B, Aaron, C (2016). Deep Learning. MIT Press

[9] Zhou, X., Wan, Q., Zhang, W., Xue, X., & Wei, Y. (2016). Model-based deep hand pose estimation. *International Joint Conference on Artificial Intelligence*, 2421–2427. https://yichenwei.github.io/publications/IJCAI16\_DeepHandModel.pdf?utm\_source=chatgpt.com

[10] Isaac, J. H. R., Manivannan, M., & Ravindran, B. (2021). Corrective filter based on kinematics of human hand for pose estimation. *Frontiers in Virtual Reality*, *2*, 663618. https://doi.org/10.3389/frvir.2021.663618

[11] Rahman, M. M., Uzzaman, A., Khatun, F., Aktaruzzaman, & Siddique, N. (2025). A comparative study of advanced technologies and methods in hand gesture analysis and recognition systems. *Expert Systems with Applications*, *266*, 125929. https://doi.org/10.1016/j.eswa.2024.125929