New Zealand Sign Language Recognition Using A Bidirectional Long Short-Term Memory Network

Daniel McGregor
Department of Mechatronics Engineering
University of Canterbury
Christchurch, New Zealand
dmc270@uclive.ac.nz

Richard Green
Department of Computer Science and Software Engineering
University of Canterbury
Christchurch, New Zealand
richard.green@canterbury.ac.nz

***Abstract* – This paper proposes a method of Sign Language Recognition (SLR) encompassing the New Zealand Sign Language (NZSL). The proposed method uses a Bidirectional Long Short-Term Memory (BiLSTM) network to train a dataset comprised of sequences of 20 images. The images in the dataset are first preprocessed by applying Haar cascading, colour segmentation, contour detection, canny edge detection and morphological dilation. A BiLSTM network is then used to train the data. The model was trained on a subset of the NZSL alphabet to determine the viability of this method. The method presented an average accuracy of 92.04% when predicting NZSL in real-time with an average frame rate of 10 Hz.**

I. INTRODUCTION

New Zealand Sign Language (NZSL) is one of New Zealand's three official languages and is crucial for communication within the deaf population. In New Zealand, approximately 4,500 deaf individuals rely on NZSL, yet its public recognition is limited, with only about 20,000 individuals using NZSL [1]. There is a significantly larger population that is near-deaf that would benefit from the use of sign language. The challenge lies in the scarcity of individuals learning this language due to the small deaf community in New Zealand. Consequently, there is a pressing need for alternative translation methods to facilitate the inclusion of the deaf population in society.

Given the scarcity of deaf individuals in New Zealand, there exists little motivation for the broader public to acquire proficiency in NZSL. Consequently, alternative translation methods become imperative to integrate the deaf population into society. This paper presents a novel approach to interpreting sign language, featuring near-real-time detection capabilities and the ability to interpret both static and dynamic signs. The selection of New Zealand Sign Language for this study is grounded in its relevance to the local context and its added complexity, distinguishing it from some other common Sign Languages like American Sign Language, which predominantly employs a single hand.

This paper proposes a method for sign language recognition (SLR) which holds substantial promise for facilitating communication between those unfamiliar with NZSL and the deaf community. This tool, capable of interpreting the intricate gestures of NZSL, addresses a critical need for deaf individuals to engage more seamlessly with society. This method of a sign language interpreter would provide useful groups such as emergency responders.

The methodology employed in this research leveraged a comprehensive database of images. These images served as the foundation for extracting pertinent features, subsequently integrated into a deep learning model.

II. BACKGROUND

Prior research has explored several methodologies for the detection and interpretation of sign language, often tailored to the nuances of different sign languages, each presenting its unique set of challenges. While some of the earlier studies aimed at developing a general sign language interpreter, the inherent differences among sign languages, such as the use of multiple hands in New Zealand Sign Language and the reliance on a single hand in American Sign Language, necessitate specialized approaches for effective interpretation. Each of these studies has its advantages and limitations.

A. Convolutional Neural network

Most Sign Language Recognition (SLR) approaches have leaned towards employing Convolutional Neural Networks (CNNs). The differentiation between these

approaches using CNNs lies in the preprocessing techniques applied to the dataset.

One study conducted by O. Sheta and R. Green employed Canny Edge Detection and Binary Thresholding, resulting in accuracy rates of 7.73% and 4.76%, respectively [2]. However, these methods failed to extract specific features relevant to sign language, capturing a significant amount of irrelevant information and notably decreasing overall testing accuracy. To enhance accuracy, the outline of hands could be isolated, and all other irrelevant features discarded.

In another approach by L. McNight and R. Green, image subtraction and thresholding were utilized [3], incorporating transfer learning and achieving a testing accuracy of 99.53%. Transfer learning, involving the retraining of the last layers of a model with a specific dataset using a pretrained model, exhibited promise. Transfer Learning, however, presents some challenges such as overfitting or low accuracy, which could arise if the pretrained model closely resembled or deviated too much from the target model [4].

The extraction of critical key points on a hand marked a notable advancement in capturing essential features, including the position of concealed joints [5]. While this method demonstrated superiority in certain aspects, it presented a limitation in the scarcity of feature information extracted, typically capturing only 21 key points. This limitation resulted in inaccurate predictions for similar signs, such as M and L, with accuracy dropping from 100% to 80% compared to other sign accuracies.

Despite these contributions, CNNs exhibit a significant flaw—they can only interpret static images [2] [6]. Given that SLR involves recognizing dynamic signs, a crucial aspect in sign language communication, CNNs fall short in handling sequences of images or videos representing a single sign in some cases. This limitation underscores the need for alternative models capable of interpreting the dynamic nature of sign language.

B. *Recurrent Neural Networks*

Recurrent Neural Networks (RNNs), as a fundamental type of Artificial Neural Network (ANN) [7], play a crucial role in predicting dynamic signs by leveraging sequential data. The distinguishing feature of RNNs lies in their capacity to incorporate information from earlier points in a sequence, introducing a memory element that enhances their predictive capabilities [8].

The architecture of a basic RNN is illustrated in Figure 1, where all inputs and outputs are interconnected for every neuron within the network. The recurrent connections (depicted as v) in the folded representation demonstrate the temporal relationships, and the unfolding on the right of the figure illustrates a time series of layers within the network [8].

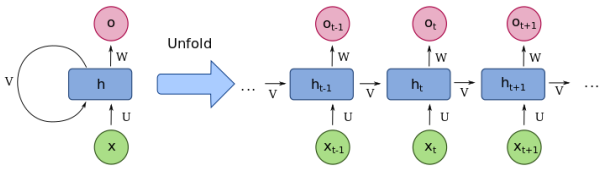

Figure 1. Basic Recurrent Neural Network [7].

However, traditional RNNs face challenges, notably the vanishing and exploding gradients problem [10]. In neural networks with n hidden layers, the multiplication of n derivatives becomes a critical factor. When these derivatives exhibit substantial magnitudes, the gradient experiences exponential growth during propagation, resulting in the exploding gradient problem. Conversely, if the derivatives are diminutive, the gradient undergoes exponential decay as it traverses the model, giving rise to the vanishing gradient problem [11].

C. *Long Term Short-Term Networks*

This led to the creation of the Long Short-Term Memory (LSTM) Network which solves the vanishing and exploding gradients problem. The architecture of the LSTM network, depicted in Figure 2, introduces a gated cell as its hidden layer. Unlike traditional Recurrent Neural Networks (RNN), an LSTM network incorporates three sigmoid gates and one tanh layer, unlike traditional RNNs which only have a single tanh layer, providing a robust solution to gradient-related issues. The output ranges from 0 to 1, where 0 signifies rejection, and 1 signifies inclusion [9].

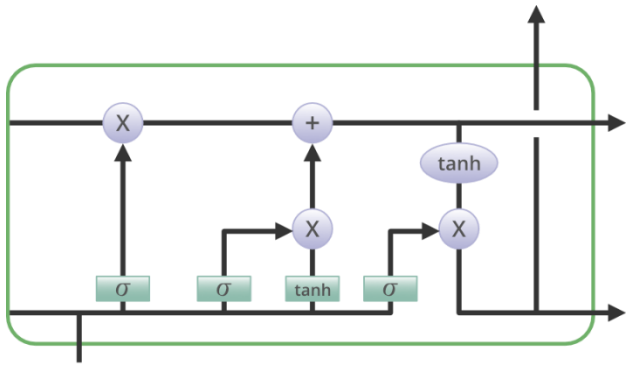

Figure 2. Structure of an LSTM network [9].

An extension of the LSTM network is the Bidirectional LSTM (BiLSTM), utilizing two LSTM networks. One network processes the input in a forward direction, while the other processes the information backward. This bidirectional approach proves advantageous by incorporating information from both the past and present, contributing to increased accuracy [11].

Studies on SLR employing LSTM networks introduce distinctive features not present in studies relying solely on Convolutional Neural Networks (CNNs). Some studies analytics the interpretation of dynamic gestures by comparing the efficacy of various models, including Gated Recurrent Unit (GRU), LSTM, and BiLSTM networks [12]. One study employs similar preprocessing techniques as those conducted for static images. This study achieved an estimated accuracy of up to 91.8%, revealing that the GRU network outperformed LSTM and BiLSTM networks on low complexity sequences, while LSTM and BiLSTM networks outperformed the GRU network on high complexity sequences.

Other studies using the LSTM network achieved accuracies of 95.2% and 99.4%, respectively [13] [14] through the means of combining a CNN and LSTM network, and specialized hardware such as the Leap Motion Controller.

In a related study by S. Masood, A. Srivastava, H. Thuwal, and M. Ahmad, a hybrid approach utilized both a CNN for spatial feature training and an RNN for temporal feature training [15]. Spatial features represent extracted features from an image or sequence of images, while temporal features capture the relationships between frames in a sequence. This integrated method achieved a higher accuracy of 95.2%, underscoring the significance of meticulous feature extraction in enhancing model accuracy.

The subsequent sections of this conference paper will delve into the implementation and evaluation of the proposed BiLSTM network for New Zealand Sign Language recognition, building upon the advancements and insights gained from the discussed methodologies.

III. METHOD

The proposed method employs a custom dataset subjected to preprocessing using advanced algorithms, surpassing previous approaches by optimizing the extraction of spatial information while selectively focusing on relevant features. This builds upon earlier studies, such as one that exclusively extracted edge features from images [2], integrating more sophisticated machine learning models than previous static recognition studies. The chosen approach centers around a BiLSTM network due to its demonstrated high accuracy in learning complex sequences [12].

A. Datasets

The dataset, crucial to the success of this research, was gathered using a custom script. Comprising 50 sequences for each of the 9 unique classes, this dataset serves as a proof-of-concept model to recognize static and dynamic gestures. The selected classes - "my," "name," "hello," "A," "B," "C," "M," "N," "X" - were chosen to address specific challenges. The classes "my," "name," and "hello" involve dynamic signs, introducing variability. Meanwhile, the visually similar classes "M" and "N" were included to examine potential overlapping predictions, providing insightful challenges for analysis within the proof-of-concept model. Figure 3 presents an image from the custom dataset, illustrating the signing of the letter "A.".

Figure 3. Example dataset image for the sign "A".

B. Data Pre-processing

Upon dataset collection, a crucial phase involves pre-processing to extract spatial features within each frame of a sequence, focusing on features relevant to sign language, specifically the shape of the hand. These methods are consistently applied to all frames in all sequences in the dataset.

i. Filtering the face

The first step involved filtering out the face in an image using the Viola-Jones algorithm [15]. The Viola-Jones algorithm employs Haar-like features defined as weighted sums of pixel differences within rectangular regions, represented mathematically in Equation 1.

$$h(x,y) = \sum_{i=1}^{n} w_i \cdot p_i(x,y) \qquad (1)$$

The integral image, expressed in equation 2,

$$II(x,y) = \sum_{i=0}^{x}\sum_{j=0}^{y} I(i,j) \quad (2)$$

efficiently calculates these features. Adaptive Boosting, an algorithm by Y. Freund and R. Schapire [17], is an ensemble learning algorithm, which is used to iteratively select and weight weak classifiers, with the final strong classifier represented in Equations 3 and 4.

$$F(x,y) = \sum_{j=1}^{N} \alpha_j \cdot H_j(x,y) \quad (3)$$

$$H_j(x,y) = \begin{cases} 1 \; if \; h_j(x,y) < T_j \\ 0 \; otherwise \end{cases} \quad (4)$$

The cascade structure is denoted by multiple stages, each comprising several weak classifiers, contributing to the overall efficiency of the algorithm for real-time object detection. Using this algorithm, a resultant image will provide a list of rectangles containing the faces in the image. These rectangles are then padded and used to black out the faces in the image as seen in Figure 4.

Figure 4. Blacked out face.

ii. Isolating the hands

With faces removed, the focus shifts to isolating the hands in the image. This process involves a Hue Saturation Value (HSV) colour conversion followed by skin colour thresholding using a mask. The mask, defined by upper and lower threshold HSV colours, outputs a binary image where colours outside these bounds are set to black.

The Suzuki-Abe algorithms, introduced by Satoshi Suzuki and Keiichi Abe in 1985, offer efficient solutions for contour tracing in binary images [18], and were used for contour finding to detect the hands in the image. The Label Following algorithm employs a pixel-wise border-following procedure to extract contours of labeled regions. Let P_i represent the i-th pixel in the contour. Equation 5 denotes its eight connected neighbors.

$$P_{i-1}, P_{i+1}, P_{i+w-1}, P_{i+w}, P_{i+w+1}, P_{i-w-1}, P_{i-w}, P_{i-w+1} \quad (5)$$

Where w is the image width. The Two-Subiteration algorithm enhances contour tracing efficiency without explicit labeling. In the first sub-iteration, contour pixels are identified, marked as "strong" or "weak," and encoded with connectivity information using binary image B, where B_c represents the binary image with contour pixels marked as "1." In the second sub-iteration, the contour is traced using the connectivity information.

The resulting contours are compared to a threshold value to filter out smaller contours with negligible information. These contours are then drawn on a black canvas and filled with the contents of the RGB image. A bitwise AND operation between the RGB image and the black canvas, preserving pixels with non-zero values in both images, results in an image as depicted in Figure 5.

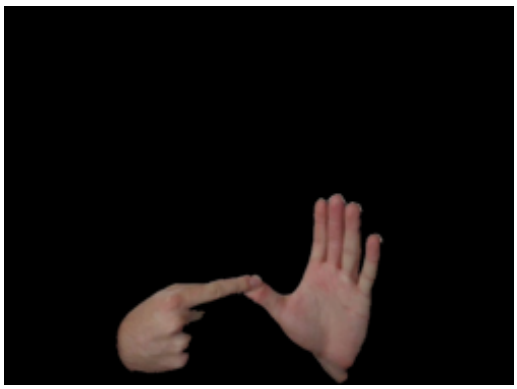

Figure 5. Hands filtered out from the original image.

iii. Extracting the hand features

The Canny Edge Detection algorithm, devised by John Canny [16], is a multi-stage process for detecting edges in digital images. This algorithm is used to outline the hand features previously found. The algorithm aims to highlight significant variations in intensity, typically indicative of object boundaries. The algorithm involves Gaussian smoothing to reduce noise, followed by gradient calculation to determine edge intensity and direction. Non-maximum suppression is applied to thin the edges, retaining only local maxima along the gradients. Subsequently, double thresholding classifies pixels as strong or weak edges. Hysteresis is then employed to link strong edges and adjacent weak edges, creating a complete edge map. The output is an image where edges, representing boundaries or features, are accentuated, so the outlines of the hands can be accurately detected.

To enhance edge intensity, a morphological operation called dilation is applied. For each pixel in the input image, the corresponding pixel in the output image is set to the maximum pixel value within a neighborhood defined by a 3 by 3 kernel [REF 18]. This dilation

operation, defined by Equation 6, ensures a clearer definition of hand outlines, preventing the omission of thin edges when downscaling the image.

$$A \oplus \mathrm{B} = B \oplus \mathrm{A} = \bigcup_{b \in B} \mathrm{A}_b \qquad (6)$$

A_b is the translation of A by b. The culmination of these processes results in an image featuring the desired spatial features – the accentuated outline of the hands, as seen in Figure 6. This image is then resized to 256px by 256px, optimizing it for seamless integration into the subsequent BiLSTM network, facilitating precise sign language recognition.

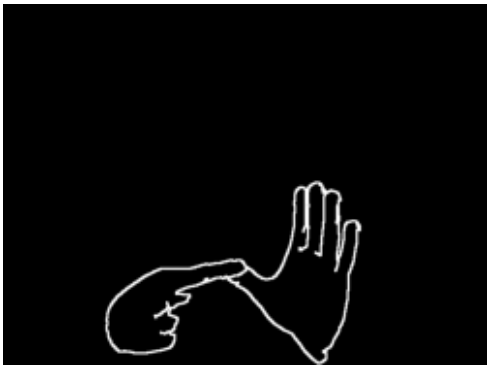

Figure 6. Outline of hands using Canny Edge Detection.

C. Bidirectional Long Short-Term Network

The proposed sign language recognition method utilizes a BiLSTM network, a type of RNN specifically designed for handling sequential data, crucial for capturing dynamic hand signs [9].

To configure the BiLSTM network, several key parameters are defined. Firstly, the input size is set to 65536, reflecting the spatial features extracted from grey-scaled and resized images (256px by 256px). The hidden state size is designated as 128, and two recurrent layers are stacked to enhance the network's depth. With nine classes in the dataset, representing different sign language gestures, the model is tailored accordingly. To prevent overfitting, a dropout probability of 20% is incorporated [19].

During each epoch of training and testing, a forward propagation method is invoked. The LSTM network is initially defined, and the dropout probability facilitates regularization by randomly setting a fraction of input units to zero at each epoch during training. The fully connected layer maps the BiLSTM layer to the outputs, with the hidden layer size doubled due to the bidirectional nature of the LSTM, iterating over both forward and backward hidden states.

This final fully connected layer is the output of the model and can be passed through a softmax function, described by Equation 7, to obtain probabilities for each sequence in the batch [20].

$$\sigma(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \qquad (7)$$

The final fully connected layer serves as the model's output, and a softmax function can be applied to obtain probabilities for each sequence in the batch.

IV. RESULTS

The results were conducted with the software and hardware specifications outlines in Table 1.

Table 1. Specifications of hardware and software used.

Operating System	Windows 11
Device Model	HP Envy x360 Ultrabook
CPU	Intel Core i7 1255U
GPU	Nvidia RTX2050
OpenCV	Version 4.9.0
Python	Version 3.11.7
Pytorch	Version 2.1.2+cu121
Camera	Logitech C920 Pro HD Webcam, 30fps, 1080p

The method's performance was evaluated by comparing its accuracy in predicting correct signs against prior studies, focusing on both static and dynamic signs.

A. Training and Testing the BiLSTM network

The BiLSTM network underwent training for 25 epochs, with Figures 7 and 8 depicting the accuracy and loss values over time. The training process took approximately 30 minutes, resulting in an overall accuracy of 92.04% and a loss of 0.0841 after 25 epochs, using a learning rate of 0.0003. Real-time prediction achieved an average frame rate of approximately 10Hz. It's important to note that these accuracies were determined using validation datasets, and variations may occur in live scenarios due to additional variables, such as lighting conditions affecting the image.

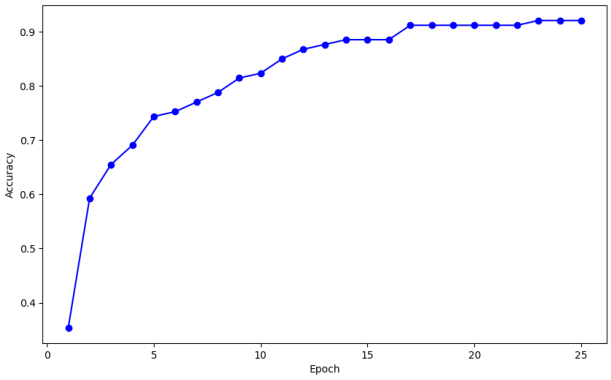

Figure 7. Accuracy of BiLSTM network.

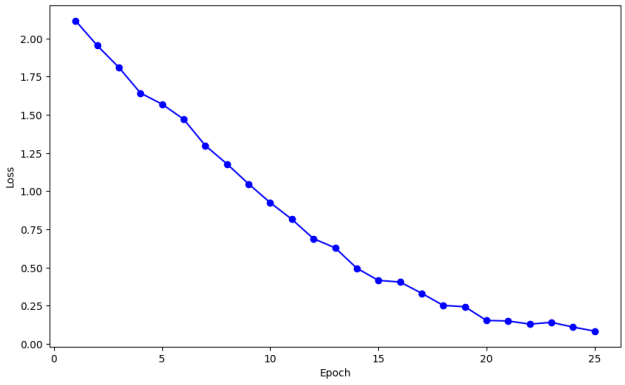

Figure 8. Loss of BiLSTM network.

The results are presented in a confusion matrix as seen in Figure 9, showcasing correct predictions and all incorrect classifications.

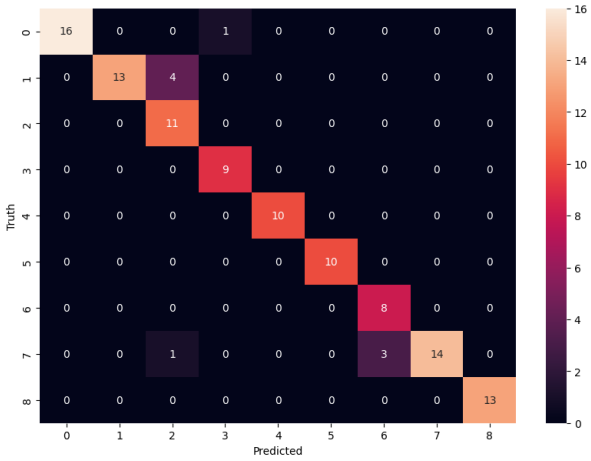

Figure 9. Confusion matrix results.

The validation dataset demonstrated high accuracy across most classifiers, with minor outliers observed for classifiers "A," "N," and one instance for "hello." The similarity between classifiers "M" and "N" was expected, while the overlap between "A" and "B" was less anticipated. Table 2 provides classifier-specific accuracy details, culminating in an overall accuracy of 92%, as previously mentioned.

Table 2. Accuracy of each classifier.

Classifier	Accuracy (%)
Hello	100
A	100
B	68.8
C	90
My	100
Name	100
M	72.7
N	100
X	100
Average	92%

B. Limitations

The proposed method has several limitations. The primary constraint lies in the reliance on colour segmentation, specifically geared towards lighter skin tones. The designated upper and lower thresholds in HSV format (0, 20, 70) and (20, 255, 255) respectively may not effectively detect darker skin tones. This limitation hinders the program's versatility, restricting its efficacy in predicting sign language across a diverse range of individuals.

Furthermore, occlusion of hands presents a challenge as spatial features collected cannot account for obscured hand regions. Integrating key point features into the extraction process becomes essential to infer joint positions accurately, thereby enhancing accuracy by introducing dissimilarity between similar classifiers that may appear indistinguishable.

The utilization of a face detection algorithm introduces another limitation. Each image must either lack a face in the frame or feature a face positioned approximately 0.5m to 4m away from the camera. Additionally, the face must be fully captured within the image for accurate detection. The preprocessing step involves concealing the face with black boxes (Figure 8), resulting in the exclusion of any hands overlapping with the face. This omission compromises the model's ability to predict results accurately in scenarios where hands and faces intersect.

Figure 8. Hand overlapping a face.

V. CONCLUSION

This paper proposed a method to recognise NZSL signs both static and dynamically. The method was designed to provide real-time feedback to the user based on the gesture detected in the video feed. The proposed method achieved a high accuracy of 92.04% with a loss rate of 0.0841.

This paper improves on previous studies by enabling the ability to recognize dynamic signs, with little compromise in accuracy. A static sign language interpreter using a CNN by L. McNight and R. Green achieved a higher accuracy of 99.53%. This accuracy is greater than that achieved by the proposed method.

Another previous study by S. Masood, A. Srivastava, H. Thuwal and M. Ahmad with the ability to recognize both static and dynamic signs achieved a prediction accuracy of 95.2% [15], once again higher than that achieved by the proposed method. The proposed method, however, still similarly compares with results from all studies mentioned sitting above 90% accuracy.

The proposed method did not require any specialized hardware unlike similar studies [13]. The use of specialized hardware allows accuracies of up to 99.44% for the entire alphabet. Another method that combined the CNN and LSTM networks also achieved a higher accuracy of 95.2%.

The combination of multiple neural networks, using a CNN for the extraction of both spatial and temporal features would be a critical consideration for future improvements to the proposed method. Several limitations were identified in the proposed method. Removing the reliance on detecting and removing the face in an image would increase the area within an image that a hand can be recognised, and therefore make the model more accessible for general use. The proposed method has susceptibility to lighting variations due to the use of colour segmentation and constraints introduced by face detection, leading to a significant area within the image where hands cannot be accurately detected. Addressing these limitations would be crucial for enhancing the robustness and applicability of the proposed method in diverse real-world scenarios.

VI. REFERENCES

[1] N. Garcia, “Loud and clear praise from Kaitāia’s deaf or hard-of-hearing community”, *NZ Herald*, Sep 2023. [Online]. Available: https://www.nzherald.co.nz/northland-age/news/loud-and-clear-praise-from-kaitaias-dhh-community/VFCD3TIEJREA3OJZT6MOOEKUSQ/ [Accessed 12 Jan 2024].

[2] O. Sheta and R. Green, “Real time NZSL detection with Convolutional Neural Networks”, Computer Vision Lab, University of Canterbury, Tech. Rep., 2022.

[3] L. McKnight and R. Green, “Using a Computer Game and Convolutional Neural Network to Teach Sign Language”, Computer Vision Lab, University of Canterbury, Tech. Rep., 2021.

[4] N. Joshi, “Exploring the limits of transfer learning”, *Allerin*, Feb 2020. [Online]. Available: https://www.allerin.com/blog/exploring-the-limits-of-transfer-learning [Accessed 22 Jan 2024].

[5] H. Duyen and R. Green, “Hand Gesture Recognition for Sign Language”, Computer Vision Lab, University of Canterbury, Tech. Rep., 2021.

[6] A. Eason and R. Green, “American Sign Language Teaching Assistant Using Hand Keypoint Detection”, Computer Vision Lab, University of Canterbury, Tech. Rep., 2022.

[7] Wikipedia Contributors, “Recurrent neural network,” Wikipedia, Dec. 03, 2018. [Online]. Available: https://en.wikipedia.org/wiki/Recurrent_neural_network [Accessed 18 Jan 2024].

[8] IBM, “What are Recurrent Neural Networks?” www.ibm.com, 2023. [Online]. Available: https://www.ibm.com/topics/recurrent-neural-networks [Accessed 20 Jan 2024].

[9] “Understanding of LSTM Networks,” GeeksforGeeks, May 10, 2020. [Online]. Available: https://www.geeksforgeeks.org/understanding-of-lstm-networks/ [Accessed 20 Jan 2024].

[10] K. Pykes, “The Vanishing/Exploding Gradient Problem in Deep Neural Networks,” Medium, May 17, 2020.

[11] E. Zvornicanin, Differences Between Bidirectional and Unidirectional LSTM, Jun. 2024. [Online]. Available: https://www.baeldung.com/cs/bidirectional-vs-unidirectional-lstm#:~:text=Advantages,LSTM%20layers%20from%20both%20directions [Accessed 21 Jan 2024].

[12] G.H. Samaan. et al. “Mediapipe’s landmarks with RNN for dynamic sign language recognition”, Electronics 11, no. 19: 3228. https://doi.org/10.3390/electronics11193228.

[13] C. K. M. Lee, K. K. H. Ng, C.-H. Chen, H. C. W. Lau, S. Y. Chung, and T. Tsoi, “American sign language recognition and training method with recurrent neural network,” Expert Systems with Applications, vol. 167, p. 114403, Apr. 2021, doi: https://doi.org/10.1016/j.eswa.2020.114403.

[14] S. Masood, A. Srivastava, H. C. Thuwal, and M. Ahmad, “Real-Time Sign Language Gesture (Word) Recognition from Video Sequences Using CNN and RNN,” Advances in Intelligent Systems and Computing, pp. 623–632, 2018, doi: https://doi.org/10.1007/978-981-10-7566-7_63.

[15] P. Viola and M. Jones, “Rapid Object Detection using a Boosted Cascade of Simple Features”, Mitsubishi Electric Research Labs, Cambridge, Tech. Rep., 2001.

[16] J. Canny, “A Computational Approach to Edge Detection”, Artificial Intelligence Laboratory, M.I.T., Tech. Rep., 1986.

[17] Y. Freund and R. E. Schapire, “A Short Introduction to Boosting,” Journal of Japanese Society for Artificial Intelligence, vol. 14, no. 5, pp. 771–780, 1999.

[18] S. Suzuki. and K. Abe, “Topological Structural Analysis of Digitized Binary Images by Border Following”, Shizuoka University, CVGIP 30 1, pp 32-46, 1985.

[19] S. M. Nacer, B. Nadia, R. Abdelghani, et al. “A novel method for bearing fault diagnosis based on BiLSTM neural networks”, Int J Adv Manuf Technol 125, 1477–1492, 2023, doi: https://doi.org/10.1007/s00170-022-10792-1.

[20] K. E. Koech, “Softmax Activation Function — How It Actually Works,” Medium, Nov. 28, 2020. [Online]. Available: https://towardsdatascience.com/softmax-activation-function-how-it-actually-works-d292d335bd78 [Accessed 29 Jan 2024].