engineering proceedings

CITESCORE 0.7

Proceeding Paper

Hand Gesture Recognition in Indian Sign Language Using Deep Learning

Harsh Kumar Vashisth, Tuhin Tarafder, Rehan Aziz, Mamta Arora and Alpana

MDPI

https://doi.org/10.3390/engproc2023059096

engineering proceedings

MDPI

Proceeding Paper

Hand Gesture Recognition in Indian Sign Language Using Deep Learning †

Harsh Kumar Vashisth, Tuhin Tarafder, Rehan Aziz, Mamta Arora * and Alpana

Department of Computer Science and Technology, Manav Rachna University, Faridabad 121010, Haryana, India; harshkv312001@gmail.com (H.K.V.); ttarafder001@gmail.com (T.T.); rhnaziz532@gmail.com (R.A.); alpana@mru.edu.in (A.)

* Correspondence: mamta@mru.edu.in

† Presented at the International Conference on Recent Advances on Science and Engineering, Dubai, United Arab Emirates, 4–5 October 2023.

Abstract: Sign languages are important for the deaf and hard-of-hearing communities, as they provide a means of communication and expression. However, many people outside of the deaf community are not familiar with sign languages, which can lead to communication barriers and exclusion. Each country and culture have its own sign language, and some countries have multiple sign languages. Indian Sign Language (ISL) is a visual language used by the deaf and hard-of-hearing community in India. It is a complete language, with its own grammar and syntax, and is used to convey information through hand gestures, facial expressions, and body language. Over time, ISL has evolved into its own distinct language, with regional variations and dialects. Recognizing hand gestures in sign languages is a challenging task due to the high variability in hand shapes, movements, and orientations. ISL uses a combination of one-handed and two-handed gestures, which makes it fundamentally different from other common sign languages like American Sign Language (ASL). This paper aims to address the communication gap between specially abled (deaf) people who can only express themselves through the Indian sign language and those who do not understand it, thereby improving accessibility and communication for sign language users. This is achieved by using and implementing Convolutional Neural Networks on our self-made dataset. This is a necessary step, as none of the existing datasets fulfills the need for real-world images. We have achieved 0.0178 loss and 99% accuracy on our dataset.

Keywords: CNN; AI; ISL; static hand gestures; deep learning

Citation: Vashisth, H.K.; Tarafder, T.; Aziz, R.; Arora, M.; A. Hand Gesture Recognition in Indian Sign Language Using Deep Learning. *Eng. Proc.* **2023**, *59*, 96. https://doi.org/10.3390/engproc2023059096

Academic Editors: Nithesh Naik, Rajiv Selvam, Pavan Hiremath, Suhas Kowshik CS and Ritesh Ramakrishna Bhat

Published: 21 December 2023

1. Introduction

Hand gestures are a convenient and easy mode of communication for humans. An application of gestures in communication is sign language. Sign language is a mode of non-verbal communication, mostly used by people with speech or hearing impairments. There are many sign language variations that are used by people to communicate. This paper has worked on the Indian Sign Language. Sign languages are a convenient and easy way to express their thoughts and emotions for people who know sign languages. But it is very difficult for people who do not understand sign language to communicate with those who use it. This requires a trained translator to be present in many situations.

A computer-based system that can interpret sign language can be very helpful in this aspect. It can be used to improve communication by helping people understand sign languages without specialized training. Over the past few years, much research has been conducted on gesture recognition and sign language recognition. But most of it focuses on other sign languages.

There are approximately 44 million people who have disabling deafness and hearing loss [1]. This is more than the population of the United States and Canada combined!

Hence, this is the motivation to proceed with the idea of developing a machine learning model to recognize sign languages in real time. This will help out a huge part of the population in terms of communication with one another.

This work addresses the following points:

1. Implementation of machine learning and deep learning algorithms used in hand gesture recognition.
2. To fine-tune the deep learning model in order to achieve maximum performance.
3. What flaws or drawbacks does this model have that turn up in applications of the algorithms? Discuss these flaws.
4. How these flaws can be corrected, if they can be corrected currently, or what can be done about them in the future.

2. Literature Survey

Several studies have been conducted in the fields of gesture recognition, face detection, and their applications in sign language recognition and facial expression. Some of our reviewed papers have been listed here. Rosalina et al. [2] have taken about 3900 raw image files to achieve the same with over 39 alphabets, numbers, and punctuation marks, in accordance with SIBI (Sistem Isyarat Bahasa Indonesia), and the accuracy turned out to be 90%. Computer vision were used to capture the image and then extract essential data from it. ANN (Artificial Neural Network) was then used to classify the images, and at last, speech recognition was used to translate the input speech in the form of NATO phonetic language and then translate it to sign language. Image processing is conducted by morphological operators, mainly erosion and dilation. The image is to be segmented. The whole process consists of four stages: image capture (static photos in RGB color space), image processing (separate background from hand, HSV range to separate the same, blurred, then dilated), feature extraction (finding contours that are basically edges of the same color range), and classification (ANN will take B/W (Black and White) images). Lastly, they translated the given letters into speech, which can be conducted using the NATO alphabets. B. Hangün et al. [3] compare the performance of functions related to image processing as implemented in the OpenCV library. Image processing requires more computational power than regular use because of mathematical operations such as matrix inversion, transposition of a matrix, matrix convolution, Fourier transform, etc. Images are taken as matrices, and as the image resolution increases, so does the matrix order. CPUs and GPUs work on different principles; CPUs operate on series processing (one task at a time) and GPUs on parallel processing (multiple tasks at the same time). Although this does not completely hold true today, it is still practically true. CUDA (Compute Unified Device Architecture) is a parallel programming model developed by NVIDIA that works on NVIDIA GPUs. Performance in this paper is based on time vs. image size for tasks such as resizing, thresholding (making the image black and white), histogram equalization (the process of adjusting bad contrast), and edge detection. OpenCV is an important AI tool. CUDA is an API made in 2007. The testing was conducted using a modest consumer-grade GTX 970 and i7 6700. Results showed GPUs are faster at each of the four tasks, especially at higher-quality images. S. Li [4] et al. performed a review of the current state of deep learning applications in facial expression recognition. According to the review, the increase in computational power and success in deep learning have made it efficient for automatic Facial Expression Recognition (FER). The two issues are overfitting (due to a lack of training data) and non-expression-related issues such as illumination, head position, occlusion, identity bias, and age. Anger, fear, disgust, happiness, sadness, surprise, and contempt are the seven basic expressions humans perceive. FER is of two types: static (1 image) and dynamic (continuous frames). In the earlier research, shallow learning was used, but now in this paper, real-time images (due to the increase in data and computational power) are used. There are many available datasets to train the model to avoid overfitting as much as possible, such as CK+, MMI, Oulu-CASIA, JAFFE, FER2013, AFEW, SFEW, etc. Image processing is similar to the previous paper, such as V&J for face alignment (GAN), CNN for augmenta-

tion, Face normalization for illumination (IS, DCT, histogram Normalization), and pose (Hasner, GAN). Face Classification using (SVM+CNN). D. M. Prasanna et al. [5] developed a real-time GUI-based Face recognition system using open-face Detection (HOG+SVM), feature extraction (HOG), and recognition as the three stages. Several use-case scenarios are mentioned here. The issues are similar to those in the previous paper. DNN (Deep Neural Network) was used for classification. O. K. Oyedotun et al. [6] proposed a model that recognizes all 24 hand gestures as present in Thomas Moeslund's gesture recognition database. Different hand gestures can look similar when viewed from different angles in a 2D image. This makes recognizing hand gestures a challenging task. Input images are converted to binary representations and then de-noised using a media filter. The segment of the image that contains hand gestures is extracted and rescaled to a 32×32 image by pattern averaging. The proposed model uses CNNs and stacked denoising auto-encoders. The applied network topology combines a convolution and its sub-sampling layer together into a layer. Each convolution layer generates convolutional feature maps. Then, feature reduction is applied to each convolutional feature map by subsampling layers. An SDAE (Stacked Denoising Auto Encoder) is used to learn more features from input images. More features can be learned by stacking more hidden layers. Data-augmentation techniques are often used to generate additional training data and prevent model overfitting. Some commonly used data augmentation techniques for images include translation, RGB jittering and horizontal flipping, spatial augmentation on each frame for video input, and temporal translations over spatial translations. F. Zhan [7] proposes a spatio-temporal data augmentation method for better generalization and the use of 2D-CNNs to extract segments of images that contain hand gestures. Images are converted to grayscale images and rescaled to 50×50. Horizontal mirroring are used for data augmentation. MSE is used as a cost function, and SGD is used as an optimization function. L. Pigou [8] applies recent advancements in deep learning like Temporal Convolutions, Residual Networks, Batch Normalization, and Exponential Linear Units for the framewise classification of gestures and sign languages in a continuous video stream. For pre-processing, input RGB images are converted to grayscale and rescaled to 128×128. Static information in consecutive frames is removed by taking the difference between the two frames. The model uses CNN adapted for classification tasks along with back propagation for recurrent architecture. Temporal convolutions and recurrence are used to work with video frames. CNNs allow for learning multiple features hierarchically instead of extracting features manually. Separate networks are used for gesture recognition and sign language recognition. Gesture Recognition uses a many-to-many configuration. Mini-batch Gradient Descent with Adam is used for optimizing parameters. Current methods used in head pose estimation require depth information in addition to RGB information. It is not feasible to obtain in-depth information in all situations, thus limiting the applications of these methods. Change in facial features is not linear with respect to change in angles, and Euler angles, which are used to represent pose angles (yaw, pitch, toll), need additional information (rotation order of axes), making it ambiguous. H.-W. Hsu et al. [9] propose several methods to overcome these shortcomings. A multi-regression loss function, an, -2. regression loss combined with an ordinal regression loss, is proposed, which can improve recognition using just RGB information. The ordinal regression loss helps solve the problem of non-linearity of facial features and pose angles by learning features that can rank the different intervals of angles. Also, angles are represented using the Quaternion number system instead of Euler angles. In this system, angles are represented using 4 angles-, q-x., q-y., q-z. and q-w.. This representation can avoid the Gimbal lock problem as it is a 4D representation of a 3D rotation. It can also be interpolated to generate smoother renderings of rotations. The methods proposed in this paper make head pose estimation much more feasible by reducing the amount of information required. J. Gupta [10] covers hand gesture recognition using various algorithms like deep learning, CNN, Morphological operation, and emoji prediction. A database was created for this model using various hand gestures to be recognized and further used to train the model and predict the emoji. This paper used a base of 11 gestures and filters to detect

gestures and then used CNN to categorize them. It includes morphological operations, contour extraction, and CNN. The database consists of 1200 images, corresponding to every 11 emoji. It uses a camera to monitor user-created real-time gestures. Future work is to add more emojis to be predicted in real-time gesture recognition. T. Mo [11] performs a review of gesture recognition and some of its key issues. Commonly used features for gestures include shape information (position, outline, etc.), geometric properties (length, distance, etc.), and binary images. For human skin recognition, commonly used color spaces are RGB (red, Green, and Blue), HSV (Hue, Saturation, and Value), and YCbCr. Gesture recognition involves two steps: classification and recognition. Commonly used methods for this purpose are the hidden Markov Model, Neural Networks, and Template Matching Method. But these methods are time- and resource-heavy. This paper proposes SVM, which is simpler and has better generalization capability. The YCbCr color space is used for human skin recognition. After performing dimensionality reduction, SVM and PCA are used for gesture recognition. In H. Muthu Mariappan et al. [12], real-time recognition is used to identify Indian Sign Language. It uses 1300 samples of images to train the model and further predict the Indian Sign Language. The FCM (Fuzzy c-means)-based real-time sign language recognition system can recognize 40 words of ISL at a time. This method is used to enhance casual communication among people with hearing disabilities. The future work of this paper is to add more words to the system for recognition.

In Rastogi et al. [13], the key idea is to enable communication between blind, deaf, and mute individuals using a sensor-enabled glove that translates hand gestures into tactile vibrations. This allows effective two-way communication of letters, words, and simple sentences between blind and deaf-mute pairs using coded vibro-tactile stimuli. Nagpal [14] proposes a portable arm-mounted device with a camera, screen, vibrating pads, and gesture recognition software. The camera captures gestures made by a mute user. The software recognizes and converts the gestures into text displayed on the screen. This allows real-time, two-way communication using a combination of gesture recognition, on-screen text, and vibrotactile feedback. Ahire et al. [15] propose a device that enables two-way communication between deaf-mute and hearing-speaking users. The device has a display, camera, microphone, and vibrational pads on one side for the deaf-mute user. The other side has a screen, speaker, microphone, and camera for the hearing user. When the hearing user speaks into the mic, speech recognition software converts it to text displayed on the deaf-mute user's screen. This concept could help bridge the communication divide between deaf-mute and hearing-speaking communities when implemented into a working prototype. Sharma et al. [16] use an ensemble model called Trbaggboost, which uses a small amount of labeled data to label unlabeled data from a new subject. It uses three sources of data: tri-axis gyroscopes, tri-axis accelerometers, and multichannel surface electromyograms.

After reviewing the related research, methods used in different stages of gesture classification were identified. After collecting the dataset, images are first preprocessed. Preprocessing involves multiple processes, including resizing, cropping, and converting images to a color format applicable to the problem statement. Some color formats commonly used are: Grayscale [2,6–8,16], RGB [9], HSV [12], and YCrCb [11,17]. A combination of one or more of these approaches can also be used to improve performance. Gupta et al. [10] Convert the original RGB images first to HSV, then to grayscale. Morph and Contour operations are applied after further processing to obtain a mask of the hand gesture. After that, image augmentation are applied to increase the variation in training data and reduce overfitting. Some common filters applied for augmenting images are shearing, zooming, horizontal flips, and vertical and horizontal shifts. Dimensionality reduction techniques like Principal Component Analysis (PCA) [11] and Histogram of Oriented Gradient (HOG) [5] can also be used to further reduce the number of features. After data augmentation, the model are then trained. Both classical machine learning models as well as deep learning models are used for this task. Some machine learning models used include Fuzzy C-Means (FCM) [12], Support Vector Machine (SVM) [5,11], etc. For

deep learning models, CNN [7,10,17] and Deep CNN (DCNN) [18] are primarily used as they are best suited for working with images. Other deep learning models used include Temporal Residual Networks [8]. As training CNN models from scratch is time- and resource-intensive, transfer learning is widely used for image classification purposes. Some common pre-trained models used for transfer learning include ResNet34 [19], ResNet50 [16], ResNet50V2 [16], XCeption [16], InceptionV3 [19], MobileNet [16], and MobileNetV2 [16].

3. Methodology

After reviewing the recently published results and their methods, the various steps that are involved in the complete process of gesture recognition have been determined and are as shown in Figure 1. The detailed information about each step is given in the subsections.

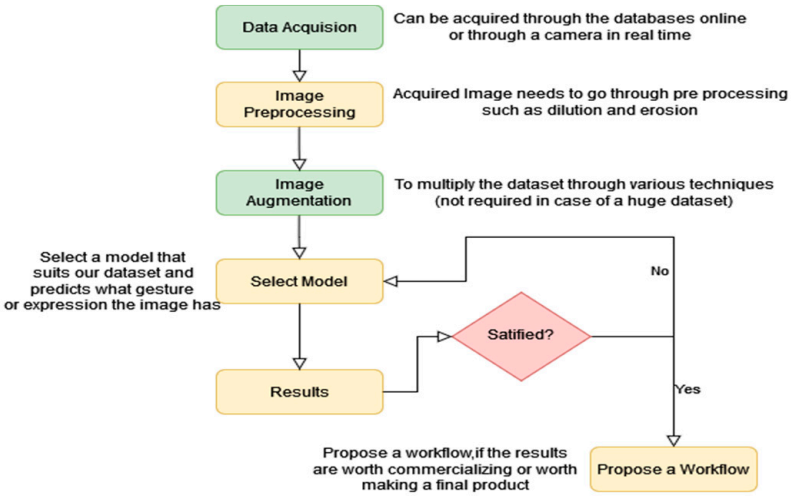

Figure 1. Process Flow Chart: Hand Gesture Recognition in the ISL Translation System.

3.1. Dataset

The dataset used in this problem is self-made, i.e., three of the four authors performed hand signs for all the images; it contains a total of 7800 images. Five burst shots for each alphabet were captured from a camera, which comprises 20 images in each burst. These images have dimensions of 4000 × 3000 pixels. This dataset is categorized into three sets named test, train, and validate, each of which contains ~5500, 1180, and 1160 images, respectively, as shown in Figure 2. There are a total of 26 alphabets, and each alphabet has 300 images. The dataset is also fairly balanced, as shown in Figure 3.

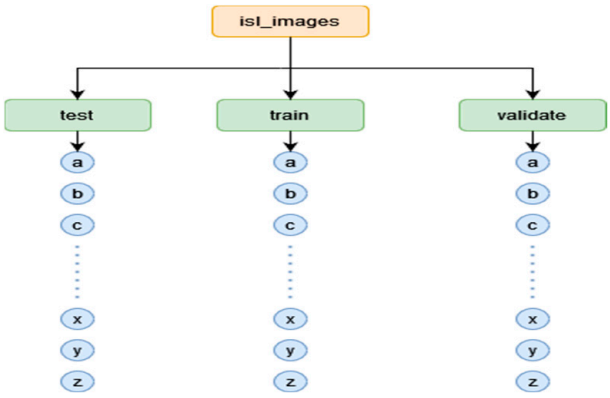

Figure 2. Dataset File Structure.

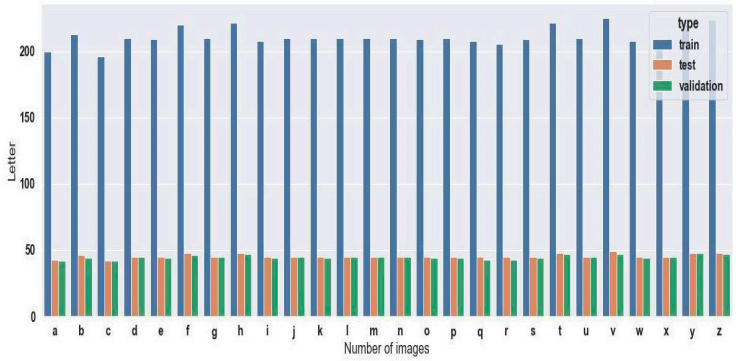

Figure 3. Number of images in each class.

3.2. Image Processing

For preprocessing, images have been rescaled to 80 by 60 pixels (to maintain the original 4:3 aspect ratio). The images, originally in RGB format, were converted to HSV format, as it better highlights hand segments in our dataset. Grayscale images for training were also used, though they performed worse. Figure 4 shows some sample images after preprocessing in HSV color space. Figure 5 shows an image preprocessed for both grayscale and HSV color spaces.

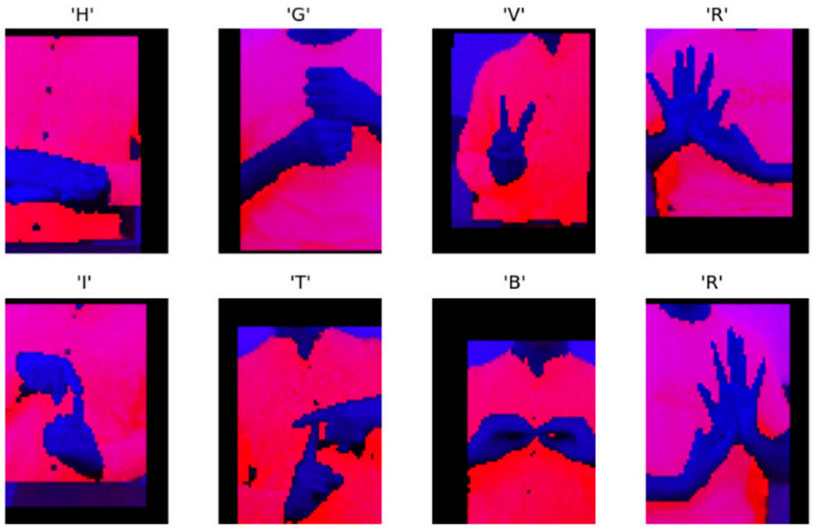

Figure 4. Some Preprocessed Images (in HSV colorspace).

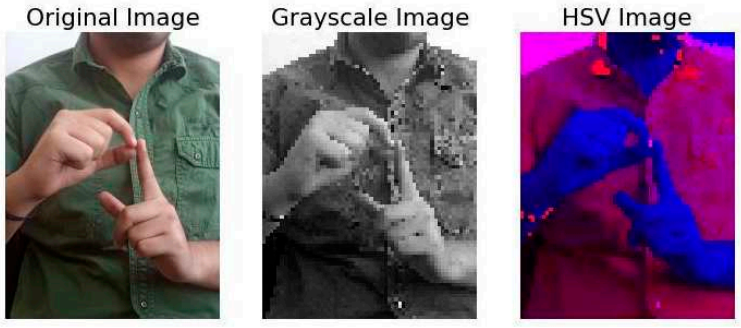

Figure 5. Preprocessing is applied to the letter 'P'.

3.3. Image Augmentation

After preprocessing, several augmentation techniques were applied to generate more training data. Some augmentation techniques applied include shearing, zooming, horizon-

tal flips, and vertical and horizontal shifts. The effects of different filters on an image are shown in Figure 6.

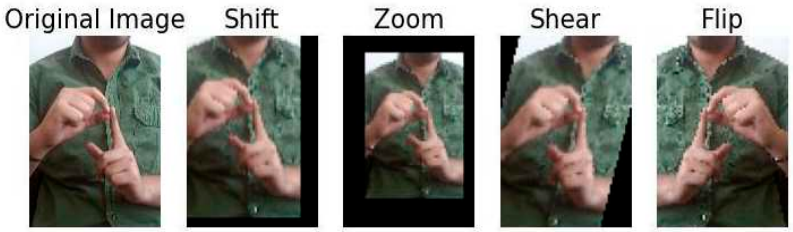

Figure 6. Augmentation filters are applied to the letter 'D'.

3.4. Model Training

This paper has used CNN [16,19,20] for our task. A Convolutional Neural Network is a feedforward neural network that has multiple layers. A CNN applies convolutions that can automatically apply preprocessing and extract features from images. So, it has the added benefit of not having to perform preprocessing and feature engineering on the image. It has multiple layers of convolutions, and each layer identifies successively complex features. The architecture of the model is shown in Figure 7.

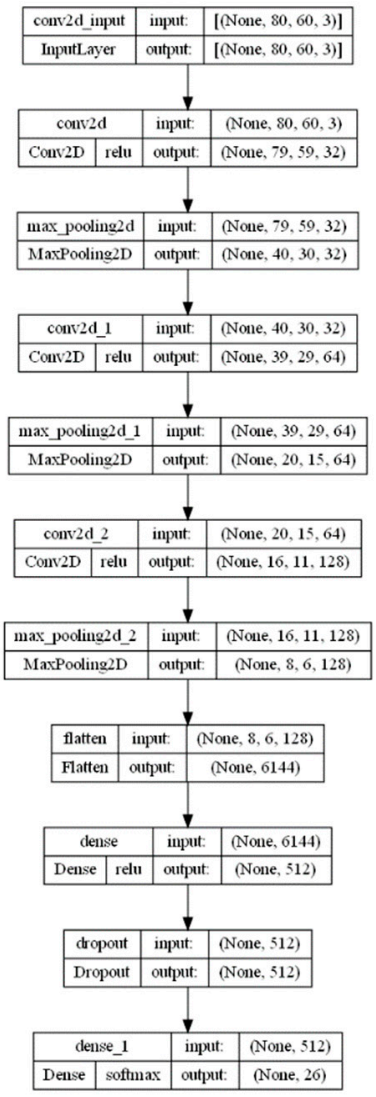

Figure 7. Model Architecture for Hand Gesture Recognition in ISL Translation.

The different layers used in the network are:

1. Conv2D: It applies a 2D convolution operation to the input data. A filter, which is a matrix whose dimensions are specified by the kernel size, is used to produce an output.
2. MaxPool2D: This is a pooling layer and is usually applied after a convolution layer. It applies a filter and selects a single value from each subregion of the specified dimension from the input data. In this case, the filter applied is Max, which selects the maximum value.
3. Flatten: It converts multi-dimensional data into a 1-D shape, i.e., flattens the data.
4. Dense: A dense layer is one where each node is connected to every node of the previous layer. In this case, connect each node to the flattened layer.
5. Dropout: The dropout layer randomly drops out nodes from its input layer at a specified rate. It is used to reduce overfitting.

All images are resized to 80 × 60 and converted to HSV color space. So, the input layer has dimensions (80, 60, 3). Models trained using HSV images provided better accuracy than grayscale images, as can be seen in Table 1. Next, 3 2D-convolution layers with ReLu as the activation function and a 2D-max pooling layer are used. Finally, a flattening layer flattens its input matrix and passes the 1D vector to a Dense layer. The output layer is a Dense layer with a SoftMax activation function and 26 nodes, corresponding to the 26 alphabets. Because our task is a multiclass classification problem, SoftMax is used, as it can convert model outputs to a probability distribution that can be easily used to select the predicted classes.

Table 1. Comparison of Hand Gesture Recognition Models for ISL Translation.

System Configuration	Optimizer	Input Size	Training Time	Testing Loss	Testing Accuracy
Google Colab GPU: Tesla K80 CPU: Intel(R) Xeon	rmsprop	(100, 75, 1) Grayscale	3 h 52 min	0.0687	0.9713
Google Colab GPU: Tesla K80 CPU: Intel(R) Xeon	rmsprop	(80, 60, 3) HSV	3 h 26 min	0.0178	0.999
Local Machine GPU: GTX 1650S CPU: Intel(R) Core i5 10400F	rmsprop	(100, 75, 1) Grayscale	2 h 24 min	0.1874	0.9383
Local Machine GPU: GTX 1650S CPU: Intel(R) Core i5 10400F	adam	(200, 150, 1) Grayscale	2 h 17 min	0.166	0.9274
Local Machine GPU: RTX 3050 CPU: Ryzen 9 5900HS	rmsprop	(100, 75, 1) Grayscale	2 h 11 min	0.025	0.9899

The model has been trained for 25 epochs. The batch size was not explicitly stated. Training was limited to 100 steps per epoch, and validation was limited to 50 steps per epoch.

3.5. Evaluating Performance

Hand gesture classification is a multi-class classification task. In our implementation, there are 26 possible outputs, all the letters of the alphabet from A to Z. Because this is a multiclass classification problem, SoftMax has been used as the activation function for the output layer. The F1 Score, Accuracy, and Confusion Matrix have been calculated to evaluate the model. The graph in Figure 8 shows the evolution of loss and accuracy over the number of epochs.

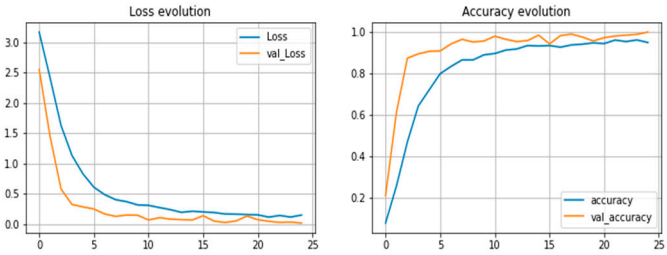

Figure 8. Evolution of loss and accuracy.

4. Results and Discussions

The model was run with different preprocessing steps and different machines, and the results are shown in Tale 1. It is observed that the model with HSV color space images as input shows the best results. Multiple configurations of hyperparameters have been tested in the models to find the one that provides optimal accuracy and loss. Doing this not only proves to be useful for further research but also completes one of the objectives stated in the introduction. Table 1 shows the performance of different models/preprocessing steps.

Table 2 shows the performance comparison of this work and existing works.

Table 2. Comparison between this paper's result and previous published results.

Paper	Task	Model	Accuracy
Muthu Mariappan et al. [12]	40 ISL words and sentences in real time	Fuzzy c-means	75%
Rosalina et al. [2]	39 ASL signs (26 alphabet letters, 10 digits, and 3 punctuation)	ANN	90%
Oyebade K. Oyedotun et al. [6]	24 ASL Hand Gestures	CNN	92.83%
Hsien-I Lin et al. [17]	7 hand gestures	CNN	99%
Gupta et al. [10]	11 hand gestures	CNN	99.6%
Proposed model	26 ISL Hand Signs	CNN	99%

The model was run with different preprocessing steps and different machines, and the results are shown below. It is observed that the model with HSV color space images as input shows the best results. Multiple configurations of hyperparameters have been tested in the models to find the one that provides optimal accuracy and loss. Doing this not only proves to be useful for further research but also completes one of the objectives stated in the introduction. Table 1 shows the performance of different models/preprocessing steps. Table 2 shows the performance comparison of this work and existing works.

From Table 1, we can conclude that using RMSprop as the optimizer, images with sizes 80 and 60 in the HSV color space provided the best performance.

From Table 2, it can be deduced that the proposed model maintains comparable accuracy while including more classes than any of the previous research.

5. Conclusions and Future Work

This paper has provided a better understanding of Indian Sign Language and applications of machine learning after getting over the hype and seeing actual results. This work should be useful for the deaf community as well, as this research hopefully strives to become a helping hand for the community. The current implementation slightly overfits the dataset, having an accuracy of 99% and a loss of 0.0178, which might be a case of overfitting and requires further work in the dataset given below. This further results in poor performance on new images. The performance of the model can be improved by taking into account hand pose/skeletons for the prediction. For future work, we aim to improve this and develop a much better implementation with aspects such as better vocabulary, a voice interface within a working client, and adding non-static signs (for instance, the letters y and j are non-static in ISL). The dataset needs to also come from more individuals with different ethnicities and backgrounds, skin tones, and varied shapes of hands and faces with their own variations of the language; furthermore, the lighting and background need

to vary to avoid bias. The data were not gathered from real practitioners of the ISL and may not be completely reflective of how a real practitioner uses the language. In addition, the model only works on static gestures; dynamic gestures need to be added in further work. There are no specific hardware requirements, and it performs reasonably well when deployed on smartphones with mediocre hardware.

Author Contributions: Conceptualization, H.K.V. and R.A.; methodology, T.T.; software, R.A.; validation, M.A., R.A., and H.K.V.; formal analysis, M.A. and A.; investigation, R.A.; resources, T.T.; data curation, R.A.; writing—original draft preparation, H.K.V.; writing—review and editing, M.A.; visualization, T.T.; supervision, M.A.; project administration, M.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available upon request from the corresponding author.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Rajput, L.; Gupta, S. Sentiment Analysis Using Latent Dirichlet Allocation for Aspect Term Extraction. *J. Comput. Mech. Manag.* **2023**, *2*, 8–13. [CrossRef]
2. Rosalina; Yusnita, L.; Hadisukmana, N.; Wahyu, R.B.; Roestam, R.; Wahyu, Y. Implementation of Real-Time Static Hand Gesture Recognition Using Artificial Neural Network. In Proceedings of the 2017 4th International Conference on Computer Applications and Information Processing Technology (CAIPT), Bali, Indonesia, 8–10 August 2017; pp. 1–6.
3. Hangün, B.; Eyecioğlu, Ö. Performance Comparison Between OpenCV Built in CPU and GPU Functions on Image Processing Operations. *Int. J. Eng. Sci. Appl.* **2017**, *1*, 34–41.
4. Li, S.; Deng, W. Deep Facial Expression Recognition: A Survey. *IEEE Trans. Affect. Comput.* **2020**, *13*, 1195–1215. [CrossRef]
5. Prasanna, D.M.; Reddy, C.G. Development of Real Time Face Recognition System Using OpenCV. *Development* **2017**, *4*, 791.
6. Oyedotun, O.K.; Khashman, A. Deep Learning in Vision-Based Static Hand Gesture Recognition. *Neural Comput. Appl.* **2017**, *28*, 3941–3951. [CrossRef]
7. Zhan, F. Hand Gesture Recognition with Convolution Neural Networks. In Proceedings of the 2019 IEEE 20th International Conference on Information Reuse and Integration for Data Science (IRI), Los Angeles, CA, USA, 30 July–1 August 2019; pp. 295–298.
8. Pigou, L.; Van Herreweghe, M.; Dambre, J. Gesture and Sign Language Recognition with Temporal Residual Networks. In Proceedings of the 2017 IEEE International Conference on Computer Vision Workshops (ICCVW), Venice, Italy, 22–29 October 2017; pp. 3086–3093.
9. Hsu, H.-W.; Wu, T.-Y.; Wan, S.; Wong, W.H.; Lee, C.-Y. QuatNet: Quaternion-Based Head Pose Estimation With Multiregression Loss. *IEEE Trans. Multimed.* **2019**, *21*, 1035–1046. [CrossRef]
10. Gupta, J. Hand Gesture Recognition for Emoji Prediction. *Int. J. Res. Appl. Sci. Eng. Technol.* **2020**, *8*, 1310–1317. [CrossRef]
11. Mo, T.; Sun, P. Research on Key Issues of Gesture Recognition for Artificial Intelligence. *Soft Comput.* **2020**, *24*, 5795–5803. [CrossRef]
12. Muthu Mariappan, H.; Gomathi, V. Real-Time Recognition of Indian Sign Language. In Proceedings of the 2019 International Conference on Computational Intelligence in Data Science (ICCIDS), Chennai, India, 21–23 February 2019; pp. 1–6.
13. Rastogi, R.; Mittal, S.; Agarwal, S. A Novel Approach for Communication among Blind, Deaf and Dumb People. In Proceedings of the 2015 2nd International Conference on Computing for Sustainable Global Development (INDIACom), New Delhi, India, 11–13 March 2015.
14. Nagpal, N. Design Issue and Proposed Implementation of Communication Aid for Deaf & Dumb People. *Int. J. Recent Innov. Trends Comput. Commun.* **2015**, *3*, 147–149.
15. Ahire, P.G.; Tilekar, K.B.; Jawake, T.A.; Warale, P.B. Two Way Communicator between Deaf and Dumb People and Normal People. In Proceedings of the 2015 International Conference on Computing Communication Control and Automation, Pune, India, 26–27 February 2015; pp. 641–644.
16. Sharma, S.; Gupta, R.; Kumar, A. Trbaggboost: An Ensemble-Based Transfer Learning Method Applied to Indian Sign Language Recognition. *J. Ambient Intell. Humaniz. Comput.* **2022**, *13*, 3527–3537. [CrossRef]
17. Kishore, C.R.; Pemula, R.; Vijaya Kumar, S.; Rao, K.P.; Chandra Sekhar, S. Deep Learning Models for Identification of COVID-19 Using CT Images. In *Proceedings of the Soft Computing: Theories and Applications*; Kumar, R., Ahn, C.W., Sharma, T.K., Verma, O.P., Agarwal, A., Eds.; Springer Nature: Singapore, 2022; pp. 577–588.

18. Lin, H.-I.; Hsu, M.-H.; Chen, W.-K. Human Hand Gesture Recognition Using a Convolution Neural Network. In Proceedings of the 2014 IEEE International Conference on Automation Science and Engineering (CASE), New Taipei, Taiwan, 18–22 August 2014; Volume 2014, pp. 1038–1043.
19. Sharma, C.M.; Tomar, K.; Mishra, R.K.; Chariar, V.M. Indian Sign Language Recognition Using Fine-Tuned Deep Transfer Learning Model. In Proceedings of the Innovations In Computer And Information Science (ICICIS), Ganzhou, China, 27–29 August 2021; pp. 62–67.
20. Arora, M.; Dhawan, S.; Singh, K. Exploring Deep Convolution Neural Networks with Transfer Learning for Transformation Zone Type Prediction in Cervical Cancer. In *Proceedings of the Soft Computing: Theories and Applications*; Pant, M., Sharma, T.K., Verma, O.P., Singla, R., Sikander, A., Eds.; Springer: Singapore, 2020; pp. 1127–1138.