Received: April 30, 2023. Revised: June 2, 2023.

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Optimised Machine Learning-based Translation of Indian Sign Language to Text

Seema Sabharwal[1]* **Priti Singla[1]**

[1]*Department of Computer Science and Engineering, Baba Mastnath University, Rohtak, Haryana, 124001, India*
* Corresponding author's Email: sabharwalseema@gmail.com

Abstract: Sign language translation through deep learning is a popular topic among researchers nowadays. It opens the doors of communication for deaf and mute people by translating sign language gestures. The translation of input sign language gestures into text is called the sign language translation system (SLTS). In this paper, optimised machine learning-based SLTS for Indian sign language (ISL) has been proposed to facilitate deaf-mute persons. Further, this paper presents a simulation analysis of the impact of the number of convolution layers, size of stride function, epochs, and activation function on the accuracy of translation of ISL gestures. An optimised ISL translation system (ISLTS) for fingerspelled alphanumeric data of 36 classes using a convolution neural network (CNN) with a novel RADAM_NORM optimiser has been proposed. The proposed system has been implemented using two datasets- the first customised ISL alphanumeric dataset has been taken from Kaggle and the second dataset has been prepared by the author consisting of 36 classes and nearly 50K images. The accuracy of the proposed ISLTS on the first dataset is 99.446 % and on the second dataset is 97.889%.

Keywords: Indian sign language, Radam_norm optimiser, ISL dataset, Sign language translation system.

1. Introduction

Machine learning is a subfield of artificial intelligence that entails the development of algorithms that allows machines to learn from data and enhance their efficacy on particular tasks instead of being explicit coding[1]. Contrary to conventional supervised algorithms, machine learning algorithms are non-parametric supervised techniques that don't presume the statistical dissemination of input feature sets. Convolution neural network (CNN) is a type of machine learning paradigm that has been suggested in the literature for the effective translation or recognition of sign language gestures. It revolutionized the computer vision domain and provided state of art results in this arena. CNNs exceptional performance is largely attributed to the brilliant design ideas by researchers and the meticulous hyperparameter value choices. Recommendation systems, Natural language processing, are the domains in which CNN is employed [2, 3]. CNN is broadly acknowledged by a multitude of parameters and requires exhaustive tuning of hyperparameters like the layer count, batch size, activation function, optimisers, pooling and epoch size[4]–[6]. Various algorithms have been used to finetune the parameters of CNN[7]. To solve this, we have proposed hypertuned machine learning-based translation for gestures of ISL into text. The in-depth contributions of this article are as follows-

- A hypertuned machine learning-based translation system for alphanumeric ISL gestures into text has been proposed.
- An alphanumeric image dataset of ISL has been made with 36 classes and nearly 50k images.
- The role of various hyperparameters such as count of convolution layers, epochs, stride and activation function has been analysed on the efficacy of the proposed system.
- A novel optimisation function RADAM_NORM has been employed to optimise the efficiency of the Indian sign language translation system (ISLTS).

The remainder of this paper is arranged as follows: section 2 sums up the literature review, section 3 elaborates on the dataset and the methodology used, section 4 reports the results and discussion and section 5 concludes and points out some future research directions.

2. Literature review

SLTS can be categorised broadly into three parts based on the modalities- hardware-based, vision-based and hybrid SLTS. In hardware-based SLTS, wearable and device based such as cameras, webcams, data gloves, Kinect, and leap motion controllers are used to input sign language gestures[8-9]. Despite their precision, they are quite expensive, bulky and cumbersome to wear in public places. Vision-based SLTS employs image-processing practices to process sign language gestures. It is more versatile than hardware-based SLTS [10]. Hybrid SLTS is the combination of hardware and vision-based techniques. Preliminary studies of SLTS focus on using various machine learning algorithms to classify sign language gestures. Several studies have been carried out in recent years to solve the issues of automatic SLTS with the use of deep learning techniques such as CNN [11].

In 2023, [12] proposed a sign language recognition system (SLRS) for the Bangla sign language using hybrid transfer learning and a random forest (RF) classifier to classify fingerspelled alphanumeric sign language gestures to text. Adaptive thresholding is applied for pre-processing of the dataset and the proposed system achieved an average accuracy of 97.33% in the case of digits and 91.67% in the case of alphabets. Open access dataset of 2080 images was used to evaluate the model. The model works better for smaller datasets and its performance is optimised by lowering the learning rate. A multitask sign language recognition framework built on CNN and K-nearest neighbour (KNN) module was proposed by [13]. The novel concept of Wireless sensing has been used for sign language recognition, although the model achieve an accuracy of 99.9% on smaller datasets but it took higher time to train the model. In 2022, [14] proposed an integrated Media Pipe optimised gated recurrent unit (MPOGRU) for the classification of 13 dynamic ISL words. Average prediction accuracy of 95% has been attained on a real-time dataset of 30 videos for each class. Further, the word-level American and Argentinian dataset of sign language was also used to validate the proposed model. However, the model had a limited dataset and recognised only 13 words, further, it has been optimised by using ELU and softsign classifier. In 2022, [15] created an original arabic alphabet phonetics dataset (AAPD) using sound recordings of 1420 persons. Further, Mel-frequency cepstral coefficient (MFCC) with Mel-bands number 20 is used for feature extraction in VGG based Arabic speech recognition model to achieve an accuracy of 95.68%. The model has been hypertuned by focussing on feature extraction techniques and the type of neural network. In addition to this, [16] proposed a transfer learning-based alphabet-level Arabic sign language recognition system with training and testing accuracy of 98% and 95% respectively. Open access dataset has been used to train and test the gestures of Arabic sign language using EfficientNetB4. Several pre-processing techniques and pre-trained architectures have been analysed in this paper to achieve optimal performance on single-handed gestures of Arabic sign language. In 2022, [17] developed a DeepCNN deep learning-based CNN model for the recognition of 24 alphabets of American sign language (ASL) using the MNIST dataset. It attains an accuracy of 99.67% with CNN having three convolution layers in 20 epochs. The training and validation loss was 7.7924 and 7.0703 respectively. The model has been hypertuned by the varying number of convolution layers and works only for single-handed American sign language recognition. In 2021, [18] proposed transfer learning based on Faster R-CNN for the recognition of gestures in Turkish sign language. A self-made alphabet-level dataset of 29 classes has been used to attain an accuracy of 99.82%. A similar background has been used for all the signers while constructing the dataset. Batch size and region based-CNN models have been analysed to improve accuracy. In 2019, [19] developed a Korean sign language translation model using Korean electronics technology institute (KETI) video dataset. 2D coordinates of human key points are estimated from the video dataset to translate sentences into text along with the sequence-to-sequence (seq2seq) model and attention mechanism. The model was related to the emergency domain and covered 419 words, and 105 sentences with a translation accuracy of 93.28%. The insufficient size of training data and a limited number of key points have been constraints on the accuracy of the proposed model. The lack of availability of standard datasets is a major hindrance in the domain of sign language translation [20]. Further, feedforward neural networks are considered as blackbox and researchers have focussed on individual or multiple hyper-tuning of parameters (such as number of layers, number of filters, size of stride, choice of pooling, choice of activation

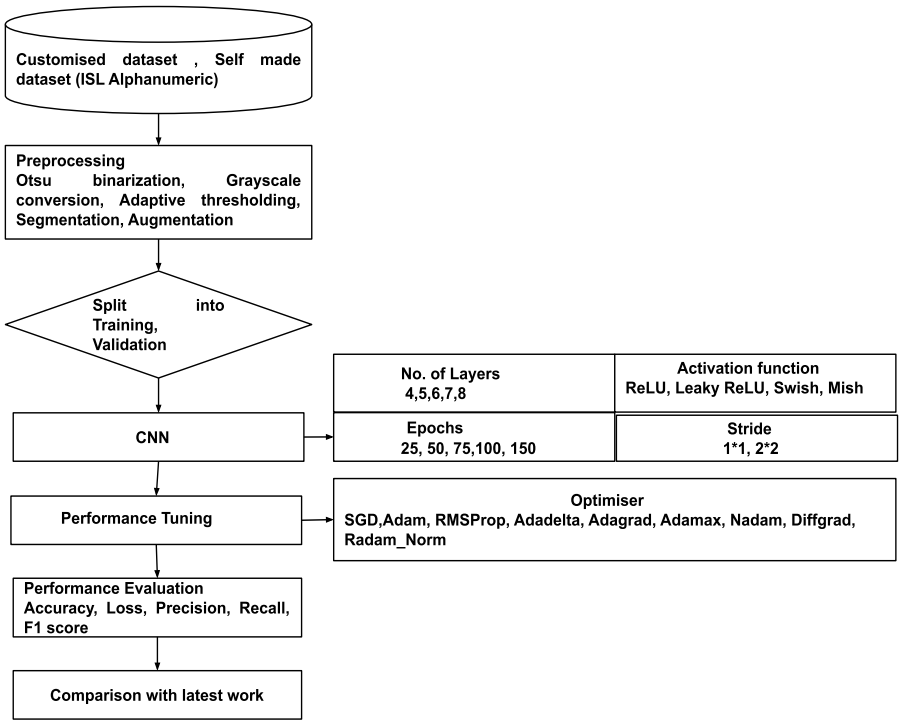


Figure. 1 Flowchart of the proposed system

function, learning rate, choice of optimiser) to have optimal performance [21]. In this paper, we have focussed on building an improvised machine learning-based Indian sign language translation system by selecting the optimal value of hyperparameters.

3. Methodology

The flowchart of our translation system has been shown in Fig. 1. The steps of the same are explained below-

3.1 Dataset

Due to the limited availability of publicly available and standard datasets, the following datasets have been employed to validate the performance of the proposed ISLTS. Firstly, customised open-access datasets from Kaggle are used for the alphabets and numbers of ISL. The dataset consists of alphanumeric data i.e., 26 alphabets (A-Z), 10 numbers (0-9). The dataset contains 36 classes and each class has 1200 images. It is referred to as dataset I. Secondly; a new image dataset is created for 26 classes of alphabets (A-Z) and 10 classes of numbers (0-9). All the images were captured by 40 persons. Different pictures per class have been taken with the signer's hand in various positions and angles. The dataset consists of nearly

Figure. 2 Proposed ISL dataset II

50K images. A sample of our dataset has been shown in Fig. 2. This dataset has been referred to as dataset II. Different backgrounds have been considered while constructing the dataset to ensure diversity and involve nonmanual features.

3.2 Data pre-processing and feature extraction

The image is transformed to a grayscale and binarized. Gaussian blur has been applied to each

input image along with Canny edge detection and adaptive thresholding. The dimensions of the input image have been reduced to 98*98 after hand detection, thresholding and segmentation. The dataset is divided into 80:20 ratios for the training and validation dataset. Data augmentation operations such as rotation, height shift, and width shift have been applied to prevent overfitting and increase the dataset.

3.3 Classification

CNNs are a particular class of machine learning algorithms that are feedforward neural networks employed in the recognition and processing of images and videos [22]. It consists of mainly three tiers-convolution, pooling and a fully connected layer. These layers are further made up of several hidden layers, a fully connected layer and a final output layer. Convolution and pooling layers are utilised for feature extraction and size reduction. The stride parameter is the extent of the shift among the application of the filter to the source image and has a default value of 1. It is uniform in height and width aspects. The role of the size of the stride function has also been analysed in this paper with the variation in several epochs. In padding, extra zeroes are added to the input matrix to maintain uniformity with output dimensions. In this paper, we have analysed the impact of the number of layers, activation function, epoch, size of stride function and optimiser on the performance of ISLTS.

The choice of hidden layers plays a vital part in the efficacy of ISLTS, as poor design may lead to overfitting and underfitting. As the count of the number of layers increases, it will lead to complexity, and few layers can't process huge image and video data. So, it is a very difficult job to decide the exact number of layers for optimal performance. In this paper, the count of convolution layers is varied to obtain maximum performance.

The term activation function symbolises the attributes of stimulated neurons that can be preserved and mapped out by a nonlinear function. It is used to address nonlinear issues. In this paper, we have compared the working of four standard activation functions ReLU, Leaky ReLU, Swish and Mish.

3.4 Finetuning

In CNN different optimisers are used to enhance the precision of the overall system. In this section, we will discuss nine states of art optimisers. Table 1. describes various notations and their description which are used for the proposed system.

Table 1. Notation list

Symbol	Description
Θ_{t+1}	Parameters (weight/bias) at time t+1
α	Learning rate
L	Loss function
$(\frac{\partial L}{\partial \Theta_t})$ or g	Gradient
ε	Small positive number
T	Iteration count
β	Decay rate
E	exponentially decaying weighted average
υ_t	Variance of the gradient at time t
m_t	Mean of the gradient at time t
$\widehat{\upsilon_{t,\iota}}$	Variance with corrected bias at time t for i^{th} parameter
$\widehat{m_{t,\iota}}$	Mean with bias correction at time t for i^{th} parameter
$\xi_{t,i}$	diffGrad friction coefficient for the i^{th} parameter at the t^{th} iteration
g_{norm}	Normalised gradient value
ρ_t	Degree of freedom (Length of approximated simple moving average)

3.4.1. Stochastic gradient descent (SGD)

It is also known as online training because all the parameters are updated as per each training image and with the same learning rate. It has slow convergence because a single record is updated in one iteration of forward and backward propagation. The training images are selected randomly to update trainable parameters i.e., weights and biases. Due to significant variance in the selected images, these modifications cause major perturbations in loss function thereby generating noise in the training phase. The weights and biases are updated based on Eq. (1).

$$\Theta_{t+1} = \Theta_t - \alpha(\frac{\partial L}{\partial \Theta_t}) \quad (1)$$

Θ_t is the value of the parameter (weight/bias) at time t, α is the learning rate, ∂L is the gradient of the loss function, $\partial\Theta_t$ is the gradient of the parameter for the selected image. The drawbacks of SGD include slow gradient, careful initialisation, sensitivity to learning rate, noisy gradient, gradient-dependent update rule and underfitting in deep architectures [1]. To deal with noise, smoothening is performed and SGD with momentum is used.

3.4.2. Adagrad

It adjusts the learning rate of each weight automatically based on previous gradient data as

opposed to previous optimisation algorithms in which the learning rate was fixed and deals with the problem of exploding gradients. Initially, the learning rate is high and then it keeps on decreasing leading to faster convergence. The equations of the parameter are shown using Eq. (2) and (3).

$$\theta_{t+1}^{i} = \theta_{t}^{i} - \alpha_t(\frac{\partial L}{\partial \theta_t}) \tag{2}$$

$$\alpha_t = \frac{\alpha}{\sqrt{\sum_{i=1}^{t}(\frac{\delta L}{\delta \theta_i})^2 + \epsilon}} \tag{3}$$

where α_t is the dynamic learning rate, t is the iteration count and ε is a small positive number.

It shows better performance with sparse data but creates problems while dealing with very deep networks or with non-convex loss functions and has premature convergence.

3.4.3. Adadelta

It is the modified version of the adagrad optimiser that resolves the problem of a relatively small learning rate in the subsequent training phases by customising it to past gradient changes and less memory requirement. In this type of optimisation algorithm, the sum of the square gradients is replaced with the exponential decaying average of the squared gradients. The parameters are updated as per Eqs. (4), (5) and (6).

$$\theta_{t+1}^{i} = \theta_{t}^{i} - \alpha_t(\frac{\partial L}{\partial \theta_t}) \tag{4}$$

$$\alpha_t = \frac{\alpha}{\sqrt{v_t + \epsilon}}, v_t = E[g^2]_t \tag{5}$$

$$v_t = \beta v_{t-1} + (1-\beta)(\frac{\partial L}{\partial \theta_t}) \tag{6}$$

where E is the exponentially decaying weighted average initialised to zero, β is the decay rate, and g is the gradient. It calculates per parameter adaptive learning rate. It requires much memory to store the additional moving average of updated learning rates It is similar to RMSProp except for the exponentially decaying average of squared parameter updates. It suffers from the problem of non-convex optimisation, slow convergence rate near minimum and computationally expensive.

3.4.4. Root mean square propagation (RMSProp)

It is a type of adaptive learning rate optimisation algorithm which is used when gradients of various parameters change widely in magnitude. It requires less memory as compared to Adadelta because it stores the moving average of squared gradients. These parameters are modified with the help of the moving average of the squared gradient, as shown in Eqs. (7) to (9).

$$\theta_{t+1} = \theta_t - \alpha_t(\frac{\partial L}{\partial \theta_t}) \tag{7}$$

$$\alpha_t = \frac{\alpha}{\sqrt{v_t + \epsilon}}, v_t = E[g^2]_t \tag{8}$$

$$v_t = \beta v_{t-1} + (1-\beta)(\frac{\partial L}{\partial \theta_t})^2 \tag{9}$$

3.4.5. Adaptive moment estimation (Adam)

It is the most commonly used optimiser in deep learning frameworks because it incorporates the merits of Nesterov momentum, AdaGrad and RMSProp optimisers. In this, features of momentum with dynamic learning rate are combined. It is an SGD approach based on the adaptive assessment of first-order and second-order moments. It is well suited for non-stationary objectives and sparse gradients. The equation of this is given by Eqs. (10) to (12).

$$\theta_{t+1,i} = \theta_{t,i} - \frac{\alpha_t \times \widehat{m_{t,i}}}{\sqrt{\widehat{v_{t,i}} + \epsilon}} \tag{10}$$

$$m_t = \beta_1 m_{t-1} + (1-\beta_1)(\frac{\partial L}{\partial \theta_t}) \tag{11}$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2)(\frac{\partial L}{\partial \theta_t})^2 \tag{12}$$

where m_t , v_t are the mean and variance of the gradient to include the features of momentum and dynamic learning rate[23]. β_1 and β_2 are decay rates for the first and second moments. To solve the problem of large step size, bias correction is introduced in the first and second moments. So, Eqs (11) and (12) with mean and variance with bias correction can be written as Eq. (13).

$$\widehat{m_{t,i}} = \frac{m_{t,i}}{(1-\beta_1^t)}, \quad \widehat{v_{t,i}} = \frac{v_{t,i}}{(1-\beta_2^t)} \tag{13}$$

3.4.6. Adamax

It is a modified version of the Adam optimiser which deals with highly fluctuating gradient values. In this optimisation algorithm, the maximum of the past gradient is used to calculate the value of weight and bias parameters. The parameters are evaluated using Eq. (14).

$$\theta_t^i = \theta_{t-1}^i - \frac{\alpha}{\upsilon_t+\epsilon}.\widehat{m_t} \quad (14)$$

where $\upsilon_t = max(\beta_2.\upsilon_{t-1}, \left|\frac{\partial L}{\partial \theta_t}\right|)$

$$\widehat{m_t} = \frac{m_t}{1-\beta_1^t}\ ,\ \ m_t = \beta_1 m_{t-1} + (1-\beta_1)(\frac{\partial L}{\partial \theta_t})$$

3.4.7. Nesterov accelerated adaptive moment estimation (Nadam)

Another variation of Adam, where Nesterov momentum is used to have faster convergence and using the gradient at the intended next place serves as the foundation for an update. The parameter update rule is given using Eq. (15).

$$\theta_t = \theta_{t-1} - \frac{\alpha}{\sqrt{\widehat{\upsilon_t}+\epsilon}}\left(\beta_1\widehat{m}_t + \frac{(1-\beta_t)\left(\frac{\partial L}{\partial \theta_t}\right)}{1-\beta_1^t}\right) \quad (15)$$

3.4.8. DiffGrad

It is a type of gradient descent optimisation method in which the learning rate for each weight is calculated dynamically by considering the first and second moments of the gradient[24]. In this, parameters are updated using Eq. (16).

$$\theta_{t+1,i} = \theta_{t,i} - \frac{\alpha_t \times \xi_{t,i} \times \widehat{m_{t,i}}}{\sqrt{\widehat{\upsilon_{t,i}}}+\in} \quad (16)$$

where $\xi_{t,i} = \frac{1}{\left(1+e^{-(|g_{t-1}-g_t|)}\right)}$

with the range [0.5,1]. $\xi_{t,i}$ is the diffGrad friction coefficient for the i^{th} parameter at t^{th} iteration to control the oscillations and slow convergence[24]. It has gained popularity in recent years due to its effective performance in the domain of deep learning.

3.4.9. Radam_norm

It is another type of adaptive SGD optimisation algorithm where gradient norm is performed using L2-norm to improve the limitations of the Adam optimisation algorithm[23]. It has been observed that normalisation will further increase the accuracy of existing optimisation algorithms and leads to fast convergence. The gradient of radam_norm is calculated using the following equations.

$$g_{norm} = L_2 Norm(g_t) \quad (17)$$

$$\theta_t = \begin{cases} \theta_{t-1} - \frac{\alpha_1 m_t}{\sqrt{\upsilon_t+\epsilon}} & if\ \rho_t >= 5 \\ \theta_{t-1} - \alpha_2 m_t & else \end{cases} \quad (18)$$

where $\alpha_1 = \frac{\alpha\sqrt{(1-\beta_2)\rho_u/\rho_d}}{(1-\beta_1^t)}$, $g_t = (\frac{\partial L}{\partial \theta_t})$

And g_{norm} is the normalised gradient value. In this method, the norms of each gradient are corrected in every iteration based on adaptive training history. It is an improved version of Rectified Adam where L2-normaliser is added. Further, this overcomes the limitation of the standard optimisation algorithm by adding normalisation.

4. Results and discussion

4.1 Dataset

To check the efficacy of the proposed system, a customised alphanumeric open-access dataset from Kaggle and a self-made dataset have been utilised.

Firstly, a customised dataset of 36 classes, approximately 1200 images per class has been sourced from the Kaggle [25], [26]. Initially, the dataset has 35 classes i.e., alphabets(A-Z) and numbers (1-9) but a class with 1200 images has been added. The resultant customised dataset has more than 45K images. This dataset has been referred to as Dataset I.

Secondly, a new dataset has been created with 40 signers. The images were captured using mobile phone cameras of varying resolutions to introduce nonlinearity and variation in the background. There were 40 signers, and 36 classes composed of 26 alphabets (A-Z) and 10 numbers (0-9) used in the creation of the dataset. Our dataset has nearly 50K images with 26 classes for the alphabet and 10 classes for number images. Other details of the dataset have been given in Table 1. This dataset has been referred to as Dataset II. The dataset has been divided into 80:20 ratios for training and validation datasets

4.2 Experimental setting

All the experiments were performed on the cloud using Google Colaboratory (Colab) Pro with 25GB RAM and 200 GB storage and Windows operating system. Pytorch, a high-level general-purpose programming language is used along with TensorFlow, Keras, Open CV, Pandas, NumPy, matplotlib etc. During the training phase of the proposed system, the following values of hyperparameters have been used as shown in Table 2.

Table 2- Initial values of the hyperparameters

Parameter	Value	Parameter	Value
α	0.001 to 0.0001	γ	0.90
β_1	0.9	Batch size	32
β_2	0.99	Pooling	Max

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 98, 98, 32)	320
max pooling2d (MaxPooling2D)	(None, 49, 49, 32)	0
dropout (Dropout)	(None, 49, 49, 32)	0
conv2d_1 (Conv2D)	(None, 49, 49, 64)	18496
max pooling2d 1 (MaxPooling 2D)	(None, 24, 24, 64)	0
dropout_1 (Dropout)	(None, 24, 24, 64)	0
conv2d_2 (Conv2D)	(None, 24, 24, 64)	36928
max pooling2d_2 (MaxPooling 2D)	(None, 12, 12, 64)	0
dropout_2 (Dropout)	(None, 12, 12, 64)	0
conv2d_3 (Conv2D)	(None, 12, 12, 128)	73856
max pooling2d_3 (MaxPooling 2D)	(None, 6, 6, 128)	0
conv2d 4 (Conv2D)	(None, 6, 6, 128)	147584
max pooling2d 4 (MaxPooling 2D)	(None, 3, 3, 128)	0
dropout 3 (Dropout)	(None, 3, 3, 128)	0
flatten (Flatten)	(None, 1152)	0
dense (Dense)	(None, 128)	147584
dense 1 (Dense)	(None, 36)	4644

Total params: 429,412
Trainable params: 429,412
Non-trainable params: 0

Figure. 3 Architecture of the proposed system

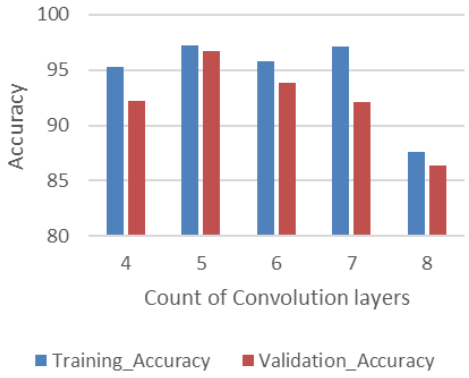
4.3 Quantitative analysis

After a comparative analysis of the count of layers, activation function, stride size, number of epochs and optimisers, CNN with 5 convolution layer, max pooling, swish activation function, 100 epochs, 1*1 stride, 32 batch size, softmax classifier and novel Radam_Norm has been selected as the optimal model. The architectural details of the proposed CNN have been shown in Fig. 3. It has 429412 parameters with 98x98 input image and L2 kernel regulariser for faster convergence, overfitting, underfitting issues and better performance.

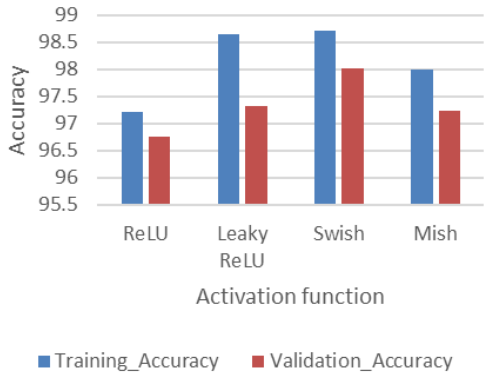
The impact of the count of convolution layers has been shown in Fig. 4 (a), 7-layer CNN with 5 convolution layers exhibited validation accuracy of 96.75% and loss of 1.2255%, which was better than all other models. The impact of the activation function has been displayed using Fig. 4 (b), and the validation accuracy rate of 7-layer CNN with ReLU, Leaky ReLU, Swish and Mish activation functions are 96.75%, 97.32%, 98.02% and 97.23% respectively. The swish activation function performed better than the remaining three activation functions. Further, the effect of the number of epochs (25,50, 75,100, 150) and stride size(1*1, 2*2) has been analysed on a 7-layer CNN with a swish activation function. The results of the same have been presented using Fig. 4 (c). The values of accuracies and loss concerning epochs and stride size have been depicted in Table 3. After finetuning of hyperparameters, the performance of the system has been enhanced using comparative analysis of the optimisers as shown in Fig. 4 (d). A novel optimiser

Table 3. Effect of Epochs, Stride Size on the proposed system

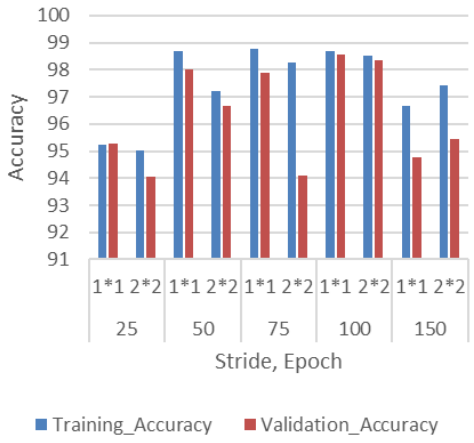
Efficacy	**Epochs**									
	25		**50**		**75**		**100**		**150**	
	1*1	**2*2**	**1*1**	**2*2**	**1*1**	**2*2**	**1*1**	**2*2**	**1*1**	**2*2**
TA	95.23	95.02	98.71	97.23	**98.77**	**98.27**	98.67	98.54	96.66	97.42
VA	95.265	94.07	98.026	96.66	**97.887**	**94.114**	98.543	98.347	94.759	95.455
TL	0.801	0.894	1.101	0.483	**0.117**	**0.136**	0.193	0.242	0.286	0.243
VL	0.9887	0.9144	1.1117	0.1878	**0.3788**	**0.3945**	0.1062	0.6095	0.2785	0.0941



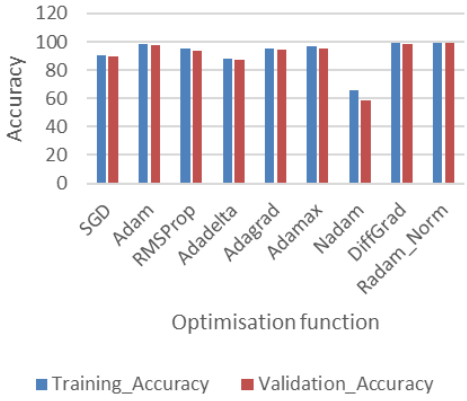
(a)



(b)



(c)



(d)

Figure. 4 Simulation analysis of proposed system: (a) Impact of layer count on accuracy, (b) Impact of activation function on accuracy, (c) Impact of epochs, stride size on accuracy, and (d) Impact of optimiser on accuracy

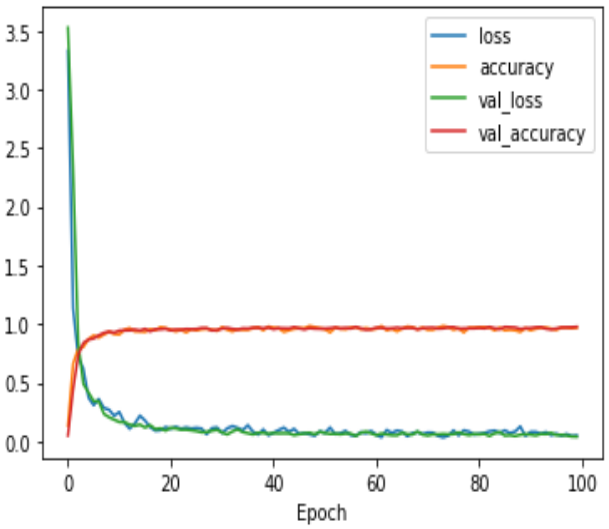


Figure. 5 Accuracy and loss curve of the proposed system

Radam_Norm has been used to achieve accuracy and loss of 98.96%, and 0.099% and its performance have been equated with other state of art deep learning

Table 4. Effect of optimiser on the proposed system

Efficacy	Optimiser								
	SGD	**Adam**	**RMSProp**	**Adadelta**	**Adagrad**	**Adamax**	**Nadam**	**DiffGrad**	**Radam_Norm**
TA	90.73	98.77	95.2	87.89	95.45	96.73	65.47	99.23	**99.446**
VA	89.99	97.88	93.75	87.56	94.21	94.85	58.34	98.76	**98.96**
TL	1.523	0.117	1.177	3.418	1.161	1.135	1.529	0.542	**0.0411**
VL	1.788	0.378	1.9987	1.4579	1.2678	1.1368	1.6354	1.6095	**0.099**

Table 5. Classification results

Dataset	Technique	Accuracy	Loss	Precision	Recall	F1-score
I	Proposed	97.889	0.0316	97.43	97.55	97.44
II		99.446	0.0411	98.87	98.01	98.43

Table 6. Performance comparison with other sign language translation models

Ref	Dataset	Technique	Modality	Accuracy
[27]	I	Multiple deep CNN models	Alphanumeric	92.85%
Ours		CNN with Radam_Norm optimiser		**97.44%**
Comparison II				
Ours	**II**	CNN with Radam_Norm optimiser	26Alphabet, 10 Number	99.46%
[28]	Own	SURF		96%
[29]	Own	CNN		89.30%

optimisers such as SGD, Adam, RMSProp, Adadelta, Adagrad, Adamax, Nadam, diffGrad with accuracy 89.99%, 97.88%, 93.75%, 87.56%, 94.21%, 94.25%, 58.34% and 98.76% respectively. The details are shown using Table 4.

4.4 Comparative analysis

The performance of the proposed automatic alphanumeric ISLTS has been demonstrated using Fig. 5, in terms of accuracy and loss of training and validation phase using Dataset II. The x-axis denotes the number of epochs and the y-axis denotes the respective value of loss and accuracy of the proposed ISLTS. The potency of the system is evaluated by employing two datasets in terms of evaluation metrics such as Precision, Recall, F1-score, Accuracy and loss. Dataset I is a customised open-access dataset of ISL gestures from Kaggle and Dataset II is a self-made dataset with images of alphanumeric data. Our proposed ISLTS demonstrated better results in the case of both datasets as shown in Table 5.

The efficacy of the proposed approach is also demonstrated by validating its performance and juxtaposing it with recent works in Table 6 and our proposed system has shown more promising results than other recent works of ISL. Due to the lack of availability of standard datasets of ISL, we have made comparisons on two criteria. Firstly, our proposed method is compared based on Dataset I, i.e., open access Kaggle dataset, in which [27] attained an accuracy of 92.85% for translating ISL gestures using pre-trained CNN models while our proposed ISLTS achieved an accuracy of 97.44%. Secondly, the comparison has been made based on the modality, [28] and [27] created in their respective alphanumeric dataset of ISL having 36 classes. [28] proposed an ISLTS using SURF and attained an accuracy of 96% and [29] proposed CNN-based ISLTS on alphanumeric dataset of their own and achieved 89.3% accuracy.

5. Conclusion and future scope

The main purpose of this paper is to build an efficient machine learning-based system for the translation of ISL gestures into text. After creating the dataset and pre-processing, different models of deep learning framework 2DCNN were analysed based on hyperparameters such as the layers count, epochs, stride size, and activation function. Further, a comparative analysis of various optimisers has been performed to enhance the performance of the proposed system. It has been concluded that the proposed system promises better results when compared with recent works in the domain of ISL translation. This study can be used by researchers to

choose optimal hyperparameters and optimisers for static alphanumeric translation of ISL gestures into text.

Conflicts of interest

The authors declare no conflict of interest.

Author contributions

Conceptualization, Seema; methodology, Seema; software, Seema; validation, Seema; formal analysis, Seema; investigation, Seema; resources, Seema; data curation, Seema; writing—original draft preparation, Seema; writing—review and editing, Seema; supervision, Priti Singla.

References

[1] P. Sharma and R. S. Anand, "A comprehensive evaluation of deep models and optimizers for Indian sign language recognition", *Graphics and Visual Computing*, Vol. 5, p. 200032, 2021, doi: 10.1016/j.gvc.2021.200032.

[2] Sitender, and S. Bawa, "Sanskrit to universal networking language EnConverter system based on deep learning and context-free grammar", *Multimedia Systems*, pp. 1–17, 2020.

[3] H. V. Chand, and J. Karthikeyan, "CNN Based Driver Drowsiness Detection System Using Emotion Analysis", *Intelligent Automation and Soft Computing*, Vol. 31, No. 2, pp. 717–728, 2022.

[4] R. G. Rajan, P. S. Rajendran, S. Smys, R. Bestak, R. Palanisamy, and I. Kotuliak "Comparative Study of Optimization Algorithm in Deep CNN-Based Model for Sign Language Recognition", in *Computer Networks and Inventive Communication Technologies*, Eds., in Lecture Notes on Data Engineering and Communications Technologies, Singapore: Springer Singapore, Vol. 75, pp. 463–471, 2022.

[5] Y. Wang, Y. Li, Y. Song, and X. Rong, "The Influence of the Activation Function in a Convolution Neural Network Model of Facial Expression Recognition", *Applied Sciences*, Vol. 10, No. 5, p. 1897, 2020.

[6] R. Nirthika, S. Manivannan, A. Ramanan, and R. Wang, "Pooling in convolutional neural networks for medical image analysis: a survey and an empirical study", *Neural Computing and Applications*, Vol. 34, No. 7, pp. 5321–5347, 2022.

[7] Y. Wang, H. Zhang, and G. Zhang, "cPSO-CNN: An efficient PSO-based algorithm for fine-tuning hyper-parameters of convolutional neural networks", *Swarm and Evolutionary Computation*, Vol. 49, pp. 114–123, 2019.

[8] T. W. Chong and B. G. Lee, "American Sign Language Recognition Using Leap Motion Controller with Machine Learning Approach", *Sensors*, Vol. 18, No. 10, p. 3554, 2018.

[9] I. A. Adeyanju, O. O. Bello, and M. A. Adegboye, "Machine learning methods for sign language recognition: A critical review and analysis", *Intelligent Systems with Applications*, Vol. 12, p. 200056, 2021.

[10] A. A. Alani, and G. Cosma, "ArSL-CNN a convolutional neural network for Arabic sign language gesture recognition", *Indonesian Journal of Electrical Engineering and Computer Science*, Vol. 22, No. 2, p. 1096, 2021.

[11] Seema, and P. Singla, A. Khanna, Z. Polkowski, and O. Castillo, "A Comprehensive Review of CNN-Based Sign Language Translation System", In: *Proc. of Data Analytics and Management*, Eds. in Lecture Notes in Networks and Systems, Singapore: Springer Nature Singapore, Vol. 572, pp. 347–362, 2023.

[12] S. Das, M. S. Imtiaz, N. H. Neom, N. Siddique, and H. Wang, "A hybrid approach for Bangla sign language recognition using deep transfer learning model with random forest classifier", *Expert Systems with Applications*, Vol. 213, pp. 118914, 2023.

[13] Z. Gao, C.C. Lee, L. Zheng, R. Zhang, and X. Xu, "A Multitask Sign Language Recognition System Using Commodity Wi-Fi", *Mobile Information Systems*, Vol. 2023, pp. 1–11, 2023.

[14] B. Subramanian, B. Olimov, S. M. Naik, S. Kim, K. H. Park, and J. Kim, "An integrated mediapipe-optimized GRU model for Indian sign language recognition", *Scientific Reports*, Vol. 12, No. 1, p. 11964, 2022.

[15] E. Almekhlafi, M. A. L. Makhlafi, E. Zhang, J. Wang, and J. Peng, "A classification benchmark for Arabic alphabet phonemes with diacritics in deep neural networks", *Computer Speech & Language*, Vol. 71, p. 101274, 2022.

[16] M. Zakariah, Y. A. Alotaibi, D. Koundal, Y. Guo, and M. M. Elahi, "Sign Language Recognition for Arabic Alphabets Using Transfer Learning Technique", *Computational Intelligence and Neuroscience*, Vol. 2022, pp. 1–15, 2022.

[17] A. Mannan, A. Abbasi, A. R. Javed, A. Ahsan, T. R. Gadekallu, and Q. Xin, "Hypertuned Deep Convolutional Neural Network for Sign Language Recognition", *Computational Intelligence and Neuroscience*, Vol. 2022, pp. 1–10, 2022.

[18] T. Yirtici and K. Yurtkan, “Regional-CNN-based enhanced Turkish sign language recognition”, *Signal Image and Video Processing*, Vol. 16, No. 5, pp. 1305–1311, 2022.

[19] S. K. Ko, C. J. Kim, H. Jung, and C. Cho, “Neural Sign Language Translation Based on Human Keypoint Estimation”, *Applied Sciences*, Vol. 9, No. 13, p. 2683, 2019.

[20] U. Nandi, A. Ghorai, M. M. Singh, C. Changdar, S. Bhakta, and R. K. Pal, “Indian sign language alphabet recognition system using CNN with diffGrad optimizer and stochastic pooling”, *Multimedia Tools and Applications*, pp. 1–22, 2022.

[21] S. H. S. Basha, S. R. Dubey, V. Pulabaigari, and S. Mukherjee, "Impact of Fully Connected Layers on Performance of Convolutional Neural Networks for Image Classification", *Neurocomputing*, Vol. 378, pp. 112-119, 2020.

[22] P. Unkule, C. Shinde, P. Saurkar, S. Agarkar, and U. Verma, “CNN based Approach for Sign Recognition in the Indian Sign language”, In: *Proc. of International Conference on Augmented Intelligence and Sustainable Systems, ICAISS 2022*, pp. 92–97, 2022.

[23] S. R. Dubey, S. K. Singh, and B. B. Chaudhuri, "AdaNorm: Adaptive Gradient Norm Correction based Optimizer for CNNs", In: *Proc. of IEEE/CVF Winter Conference on Applications of Computer Vision, WACV 2023*, pp. 5284-5293, 2023.

[24] S. R. Dubey, S. Chakraborty, S. K. Roy, S. Mukherjee, S. K. Singh, and B. B. Chaudhuri, “diffGrad: An Optimization Method for Convolutional Neural Networks”, *IEEE Trans. Neural Netw. Learning Syst.*, Vol. 31, No. 11, pp. 4500–4511, 2020.

[25] P. Arikeri, “ISL dataset Kaggle”, Accessed: Jul. Vol. 12, 2022. [Online]. Available:https://www.kaggle.com/datasets/prathumarikeri/indian-sign-language-isl

[26] P. Arikeri, “American Sign Language (ASL) Dataset”, *Kaggle*. https://www.kaggle.com/datasets/prathumarikeri/american-sign-language-09az (accessed Sep. 18, 2022).

[27] T. Kujani and V. Dhilipkumar, “Multiple Deep CNN models for Indian Sign Language translation for Person with Verbal Impairment”, *International Journal of Intelligent Systems and Applications in Engineering*, Vol. 10, No. 3, pp. 382–389, 2022.

[28] K. M. Tripathi, P. Kamat, S. Patil, R. Jayaswal, S. Ahirrao, and K. Kotecha, “Gesture-to-Text Translation Using SURF for Indian Sign Language”, *ASI*, Vol. 6, No. 2, p. 35, 2023.

[29] P. Paul and G. N. Rathna, “Real-time Indian sign language recognition”, *Summer Research Fellowship Programme of India's Science Academies*, [Online] Available: reports.ias.ac.in.