scientific reports

OPEN

An integrated mediapipe-optimized GRU model for Indian sign language recognition

Barathi Subramanian[1], Bekhzod Olimov[1], Shraddha M. Naik[1], Sangchul Kim[2], Kil-Houm Park[3] & Jeonghong Kim[1]✉

Sign language recognition is challenged by problems, such as accurate tracking of hand gestures, occlusion of hands, and high computational cost. Recently, it has benefited from advancements in deep learning techniques. However, these larger complex approaches cannot manage long-term sequential data and they are characterized by poor information processing and learning efficiency in capturing useful information. To overcome these challenges, we propose an integrated MediaPipe-optimized gated recurrent unit (MOPGRU) model for Indian sign language recognition. Specifically, we improved the update gate of the standard GRU cell by multiplying it by the reset gate to discard the redundant information from the past in one screening. By obtaining feedback from the resultant of the reset gate, additional attention is shown to the present input. Additionally, we replace the hyperbolic tangent activation in standard GRUs with exponential linear unit activation and SoftMax with Softsign activation in the output layer of the GRU cell. Thus, our proposed MOPGRU model achieved better prediction accuracy, high learning efficiency, information processing capability, and faster convergence than other sequential models.

Sign language is a vision-based interactive language with unique and complex linguistic rules. It is used by people who are hearing impaired to communicate and exchange their feelings, ideas, and thoughts using various parts of the body[1,2]. Since sign language has a unique linguistic structure, it differs from one place to another according to its geographic location[3]. Each country has developed its sign language for communication among its deaf and hard-of-hearing communities[4]. Some of the popular sign languages are American sign language (ASL) in the US[5,6], British sign language in the UK, Indian sign language (ISL) in India[7,8], Korean sign language in Korea[9]. From the World Health Organization report, approximately 500 million people worldwide suffer from hearing loss[10,11]. Because of the high prevalence of the hard-of-hearing community population, there has been an increased interest in eliminating communication obstacles faced within the hard-of-hearing community and other people with normal hearing[12].

Sign language recognition (SLR) develops an assistive system that automatically converts an input sign into its corresponding speech or text[13]. Thus, the SLR system is useful for overcoming the communication gap between hearing and nonhearing communities and creates a new path for human-computer interaction-based applications[14–18]. The major challenge to developing a continuous SLR system is finding a modeling prototype that acquires the sign gesture and its corresponding text. Starner et al.[19] developed a video-based real-time continuous SLR system using a single camera with 40 vocabulary signs of ASL sentences using a hidden Markov model (HMM) classifier. Similarly, Vogler and Metaxas[20] developed a continuous SLR system using three orthogonally positioned cameras to mitigate the problems caused by occlusion and uncontrolled movements in ASL sentences. HMM was used for the recognition process with a vocabulary of 53 signs, and the system was tested on 97 sign sentences, producing a recognition rate of 92.11% and 95.83%, respectively. From the above literature survey, it is crystal clear that all sensor and vision-based techniques are more restrictive and cost effective[21,22]. Although different techniques are available, the challenges of hand tracking[23,24], occlusion of hand movements[25], high computational cost[26], feature selection[27] and lower learning efficiency[28] still exist.

[1]School of Computer Science and Engineering, Kyungpook National University, Buk-gu, Daegu 41566, South Korea. [2]Division of Computer Engineering, Hankuk University of Foreign Studies, Seoul, South Korea. [3]School of Electronics Engineering, Kyungpook National University, Buk-gu, Daegu 41566, South Korea. ✉email: jhk@knu.ac.kr

To address these drawbacks, we proposed a MOPGRU SLR system that diminishes the problem of hand occlusion and lower learning efficiency by adjusting the output of the update gate using the reset gate integrated with an open-source framework called the MediaPipe Holistic pipeline[29] . Furthermore, we changed the activation function in the output layer and candidate memory state of each GRU cell to achieve a faster, simpler, and cost effective SLR system. The main contributions of this study are summarized as follows:

- We proposed a novel MOPGRU model that calibrate the resultant of the update gate by the reset gate, which enchances the learning process of the GRU gating unit, thereby accelerating the convergence rate, eliminating gradient depletion problem, and improving the learning efficiency.
- We replaced the hyperbolic tangent (Tanh) activation function in the candidate memory state with an ELU activation function to overcome the vanishing and exploding gradient problem, and further enhance the model. As a result, we obtained lower training time and good performance for the MOPGRU model than those of other variants. Additionally, the activation function of the output layer of each GRU cell was replaced with Softsign instead of SoftMax to reduce the computational complexity and hence, the training time of the model.

The rest of this paper is organized as follows: In "Related work" section introduces existing SLR methods and their limitations. In "Proposed methodology" section introduces and describes the proposed methodology. In "Experimental results and discussion" section presents the experimental settings, results, comparison of our method with other methods, and analyzes the model performance. Finally, "Conclusion" section presents our contributions and outlines the future work.

Related work

Previous researchers have emphasized their work on the prediction of sign language gestures to support people with hearing impairments using advanced technologies with artificial intelligence algorithms. Although much research has been conducted for SLR, there are still limitations and improvements that need to be addressed to improve the hard-of-hearing community[30]. This section presents a brief literature review of recent studies on SLR using sensor and vision-based deep learning techniques.

Sensor-based deep learning techniques.

To bridge the communication gap between the hard-of-hearing community and normal people, researchers have proposed a real-time ISL hand gesture recognition system that uses a Microsoft kinetic RGB-D camera for inputting images and applies deep learning techniques to achieve one-to-one mapping between the depth and RGB pixels on training over 45,000 RGB and depth images, while achieving a prediction accuracy of 98.81%[31]. Although the model resulted in good accuracy, it emphasized the need for a large dataset with more images to train and a high-pixel RGB camera. Like the aforementioned model, an algorithm with a support vector machine and Microsoft kinetic Xbox 360 RGB images for translating Indian sign language gestures into English text and speech with 100% prediction accuracy for the signs representing one numeric value and six ISL alphabets alone was proposed[32].

Using an expensive leap motion controller [LMC][26,30], researchers have proposed a training method for ASL with an long short-term memory (LSTM) recurrent neural network for handling a sequence of input and yields an average accuracy rate of 91.08%. This proposed model has several limitations due to the leap motion controller because the number of users affects the model accuracy. Furthermore, this method is limited to recognizing only one hand gesture. Neethu et al.[33] introduced a deep convolutional neural network (CNN) classification approach with a connected component analysis algorithm to segment the fingertips from the hand image and classify only eight different gestures into various classes with a 96.2% recognition rate. The performance was analyzed only in terms of sensitivity, accuracy, and recognition rate.

Gupta et al.[28] proposed a sensor-based multilabel classification to categorize ISL isolated signs by processing signals from sEMG and IMUs placed on both the forearms of signers in an integrated manner with some classification and categorization errors. Similarly, Salem et al.[34] proposed a real-time customize glove-based method with five-flex and one accelerometer sensor to recognize Arabic sign language gestures and display corresponding English text and audible sounds. Generally, motion gloves sign language prediction has high limitations in terms of hand tracking and is uncomfortable for users compared to vision-based methods. Like leap motion controllers, they are expensive, time-consuming, and may produce inaccurate calibrations due to wear and tear from the frequent usage of gloves.

Vision-based deep learning techniques.

Rastgoo et al.[35] proposed a real-time isolated hand SLR (IHSLR) from an RGB video[36] by combining deep learning models, singular value decomposition (SVD), and SSD with LSTM with ResNET50[37] and further with SSD, 2DCNN, 3DCNN[38], and LSTM to obtain features from the 3D hand coordinators and achieved a high accuracy of 99%[25]. The proposed model is simple and fast; however, the model is not able to recognize in case of high inter-class similarities, and in some cases there also exists some misclassification because of the high occlusion of two hands in some signs, making it difficult to predict hand signs correctly. Similarly, Chen et al.[39] proposed a three-tier network architecture with the short-term traffic prediction model built by using LSTM to simplify network management and to reduce communication overheads. Hurroo et al.[24] proposed a convolutional neural network (CNN) with the HSV color algorithm and various computer vision techniques for recognizing only 10 American Sign gesture alphabets and obtained an accuracy of 90%. Action recognition architectures constructed with 3D CNN models such as I3D[40] architecture

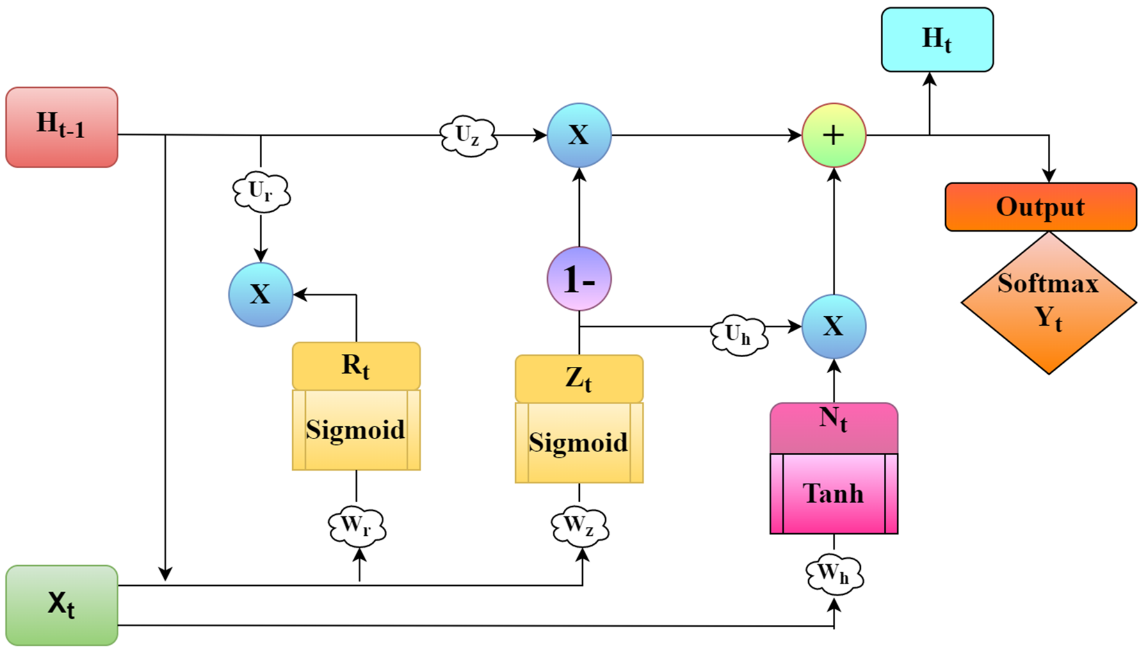

Figure 1. Structure of a standard GRU cell.

is also used for SLR task in[41]. Although this method uses low computing power, robustness is not achieved, and the prediction accuracy is lower than that of other CNN models.

Ojha et al.[42] implemented a fingerspelling sign language translator using a CNN to detect ASL and translate its corresponding text and speech in real-time. The proposed model achieved an accuracy of 95% with certain limitations; for example, when running the project, the threshold must be monitored to avoid distorted grayscale in the frames if it does not lead to resetting the histogram or looking for appropriate lighting conditions. Additionally, several extensive literature on train CNNs for continuous SLR with weakly labeled data has been reported[43]. Here, the CNN inside an iterative expectation-maximization algorithm was trained with over 1 million poorly labeled hand gesture images representing the sign language. However, moderate prediction accuracy was achieved despite using only the prerecorded pictures and videos as the input, which are unsuitable for real-time hand gesture recognition.For continuous SLR with deep learning[44], a heuristic approach for epenthesis detection to support continuous natural communication between the machine and user was proposed. Although they have reduced classification confusion, they showed good results only when tested individually with more resources needed for implementing an integrated continuous SLR system.Likewise, several studies exist where both static and dynamic gesture recognition were performed using machine learning and deep learning methods[9,45–52].

Standard GRU. Most computer vision problems require handling temporal dependencies among inputs and modeling short-term and long-term sequences. Recurrent neural networks (RNNs) are efficient in managing and processing such sequential data. Compared to traditional neural networks, RNNs focus on manipulating state neurons to learn contextual relations in and between sequential data[53]. Training RNNs is a difficult task due to several limitations and the vanishing and exploding gradient problems. GRUs were applied to solving the vanishing and exploding gradients incorporated into conventional RNNs[54,55]. Among the RNNs, the most frequently used are LSTM networks that have achieved state-of-the-art performance on various deep learning and machine learning tasks. As a variant of LSTM, GRU performs equally as an LSTM and produces good results. It enhances the configuration of the LSTM units and conjugates the three gating units to two gating units of the LSTM as update gate and reset gate. Thus, the parameters of the GRU network model are considerably less, thereby sustaining information dependency and reducing the training time. Figure 1 shows the general structure of a standard GRU cell.

From Fig. 1, at each time step t, a GRU cell takes the contents of previous hidden state H_{t-1} and present input X_t, operates them through reset and update gates, and passes the computed current state H_t to the next time step. The general formulas of a standard GRU cell are as follows:

$$R_t = \delta(W_r \cdot X_t + U_r \cdot H_{t-1} + B_r) \quad (1)$$

$$Z_t = \delta(W_z \cdot X_t + U_z \cdot H_{t-1} + B_z) \quad (2)$$

$$N_t = \tanh(W_h \cdot X_t + U_h \cdot (H_{t-1} \odot R_t) + B_h) \quad (3)$$

$$H_t = Z_t \odot N_t + (1 - Z_t) \odot H_{t-1} \quad (4)$$

$$Y_t = SoftMax((W_o * H_t) + B_o) \tag{5}$$

$$\delta(x) = \frac{1}{1+e^{-x}} \tag{6}$$

$$tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{7}$$

where R_t and Z_t denotes the resultant of the reset and update gates at time t in Eqs. (1) and (2), respectively. N_t in Eq. (3) denotes a current candidate memory value vector that is computed from a hyperbolic tangent activation function (*tanh*), represented in Eq. (7) at time t. H_t in Eq. (4) indicates the resultant of the standard GRU unit at time $t-1$, and computed as a linear interpolation of previous states H_{t-1} and N_t using the result from Z_t in Eq. (2). The Sigmoid activation (δ) in Eq. (6) is applied to both R_t and Z_t gates to scale the values within 0 and 1. $\odot$ denotes the Hadamard product which is nothing but element-wise multiplication. X_t is the current input fed into the network at time t. W_r, W_z and W_h are trainable weights of feed-forward connections, whereas U_r, U_z and U_h are weights of the recurrent connections. B_r, B_z and B_h are bias vectors. Y_t indicates the resultant of the GRU model at time t, which gives the detected result using the SoftMax activation function, and W_o, B_o represents the weight and bias of H_t. The error at each time step is calculated using the predicted output $\hat{Y}_t$ at each time step and the actual output Y_t at each time step and it is given by:

$$E_t = -Y_t log\left(\hat{Y}_t\right) \tag{8}$$

$$E = \sum_t E_t \implies E = \sum_t -Y_t log\left(\hat{Y}_t\right) \tag{9}$$

And the total error is calculated by summing up the errors at all time steps, represented in Eq. (9). From the above formulas, the GRU model accomplishes long-distance preservation of valuable particulars by reducing the number of gating units, continuously disposing of unwanted particulars, and using the hidden state to store information dependencies. Although GRU maintains a long-term information dependency, it has a slow convergence rate and low learning efficiency. Therefore, we proposed an optimized GRU that uses a reset gate to optimize the learning structure of GRU and enhance the learning and prediction accuracy.

MediaPipe. MediaPipe is an open-source framework with a hybrid platform that creates pipelines for processing perceptual data, such as images, videos, and audio. It is an extensive approach employed with ML for hand tracking and gesture recognition in real-time. It provides more hand and finger tracking solutions by accurately detecting the sign gestures. Specifically, we employed a MediaPipe Holistic pipeline to obtain the landmarks from the face, hands, and body pose. Figure 2 clearly outlines the overall functionality.

MediaPipe holistic pose landmarks. The MediaPipe Holistic body pose model infers approximately 33 3D landmarks consisting of x, y, and z coordinates on the body from the input image or video using its BlazePose detector and locates the person/pose regions of interest (ROI) within the frame. Using the ROI-cropped frame as input, the pose landmark and division masks within the ROI detect poses successively. Thus, it accurately localizes more key points and suitably fits SLR.

MediaPipe holistic hand landmarks. MediaPipe Holistic hands infer approximately 21 3D hand landmarks consisting of x, y, and z coordinates in just a single frame and produce the desired output by combining two models: the palm detection model and the hand keypoint localization model. Initially, the model was employed with a single-shot detector called Blaze Palm. This detector supports the MediaPipe to reduce the time complexity of palm detection given a large dataset of hand sizes in the input image. This model works on the entire image and returns a focused bounding box that highlights the rigid parts, such as palm and fist, for palm detection rather than concentrating on unnecessary objects. Then, the model uses the palm detection output to perform hand keypoint localization. This produced three possible outputs as follows:

- 21 hand knuckle points in a 2D or 3D space.
- Hand flag showing the probability of hand presence in the input image.
- Binary classification of left and right hand.

MediaPipe holistic face landmarks. The MediaPipe face mesh is a face geometry solution that calculates 468 3D face landmarks in real-time with a single input camera and not a depth sensor. It works based on two deep neural network models, a detector that computes and operates face locations on a full image and a 3D face landmark model that operates on the computed locations that predict approximate surface geometry using regression. With accurate cropping of the face, data augmentation processes, such as rotation, scaling, and translation are reduced, allowing the network to focus more on coordinate prediction accuracy.

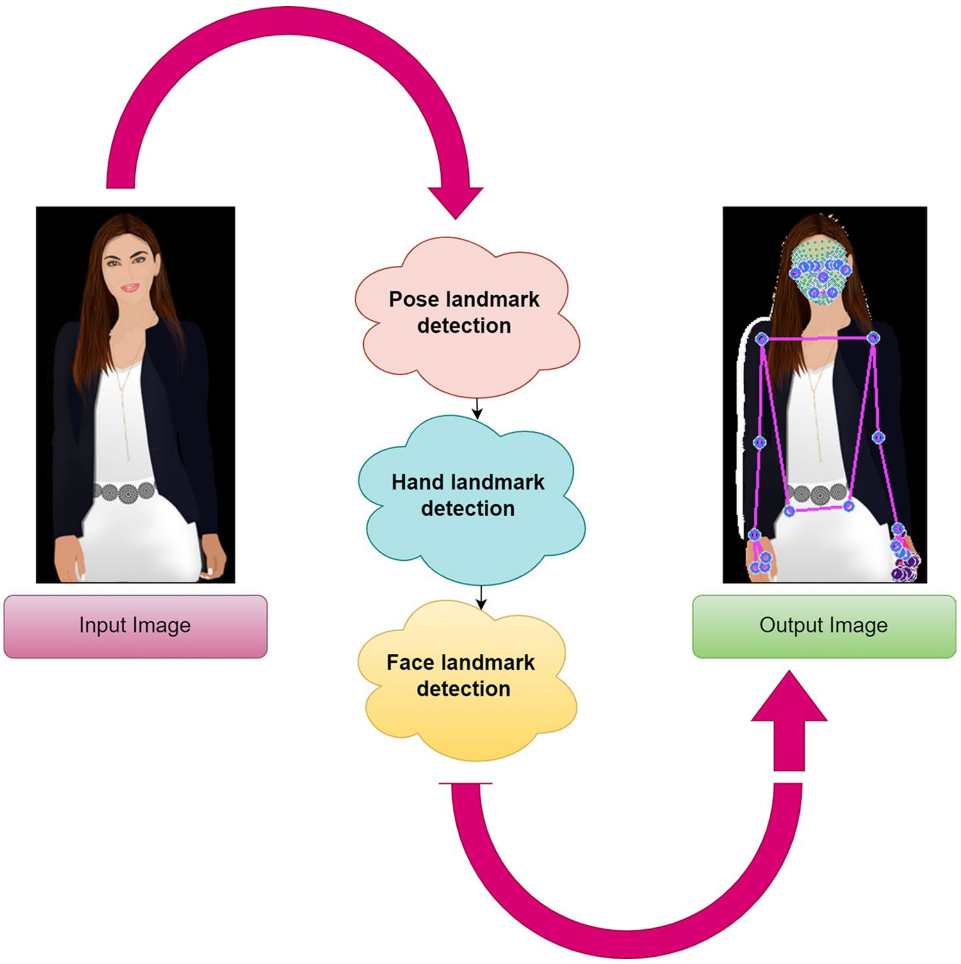

Figure 2. Overview of MediaPipe holistic.

Proposed methodology

To accurately recognize the sign gestures and translate them into text, our proposed method comprises three stages: data preprocessing and feature extraction, data cleaning and labelling and gesture recognition. Data preprocessing and feature extraction are carried over by the MediaPipe framework. Here, features from the face, hands, and body are extracted as keypoints and landmarks using built-in data augmentation techniques from sequence of input frames taken from a web camera. In stage 2, the extracted keypoints from stage 1 are saved in a file to identify and remove the null entries from the data, after which data labelling follows. In stage 3, the cleaned and labelled gestures are trained and classified by our MOPGRU model for ISL recognition with the translated sign gestures in the form of text on the screen. Figure 3 shows a general overview of the proposed architecture for an SLR system. For further understanding, the three stages of the proposed methodology are elaborately discussed below.

Stage 1: data preprocessing and feature extraction.

For data preprocessing and feature extraction from the image, we applied a multistage pipeline from MediaPipe, called MediaPipe Holistic. For each input frame from the web camera, the MediaPipe Holistic handled individual models for the hands, face, and pose components using a region-appropriate image resolution. The workflow of stage 1 is briefly discussed below:

- The human pose and subsequent landmark model were estimated using BlazePose's pose detector. Then, three ROI crops for the face and hands (2×) were derived from the inferred pose landmarks, and a recrop was employed to improve the ROI.
- Next, the corresponding landmarks were estimated. To achieve this, the full-resolution input coordinates were cropped to the ROIs for task-specific hand and face models.
- Finally, all landmarks were combined to yield the full 540+ landmarks.

Stage 2: data cleaning and labelling.

After stage 1, the extracted features, that is, the landmark points ($21 * 3 + 21 * 3 + 33 * 4 + 468 * 3 = 1662$) per frame are flattened, concatenated and stored in a file to check and remove any null entries from the data. Data cleaning is important since it prevents failed detection of

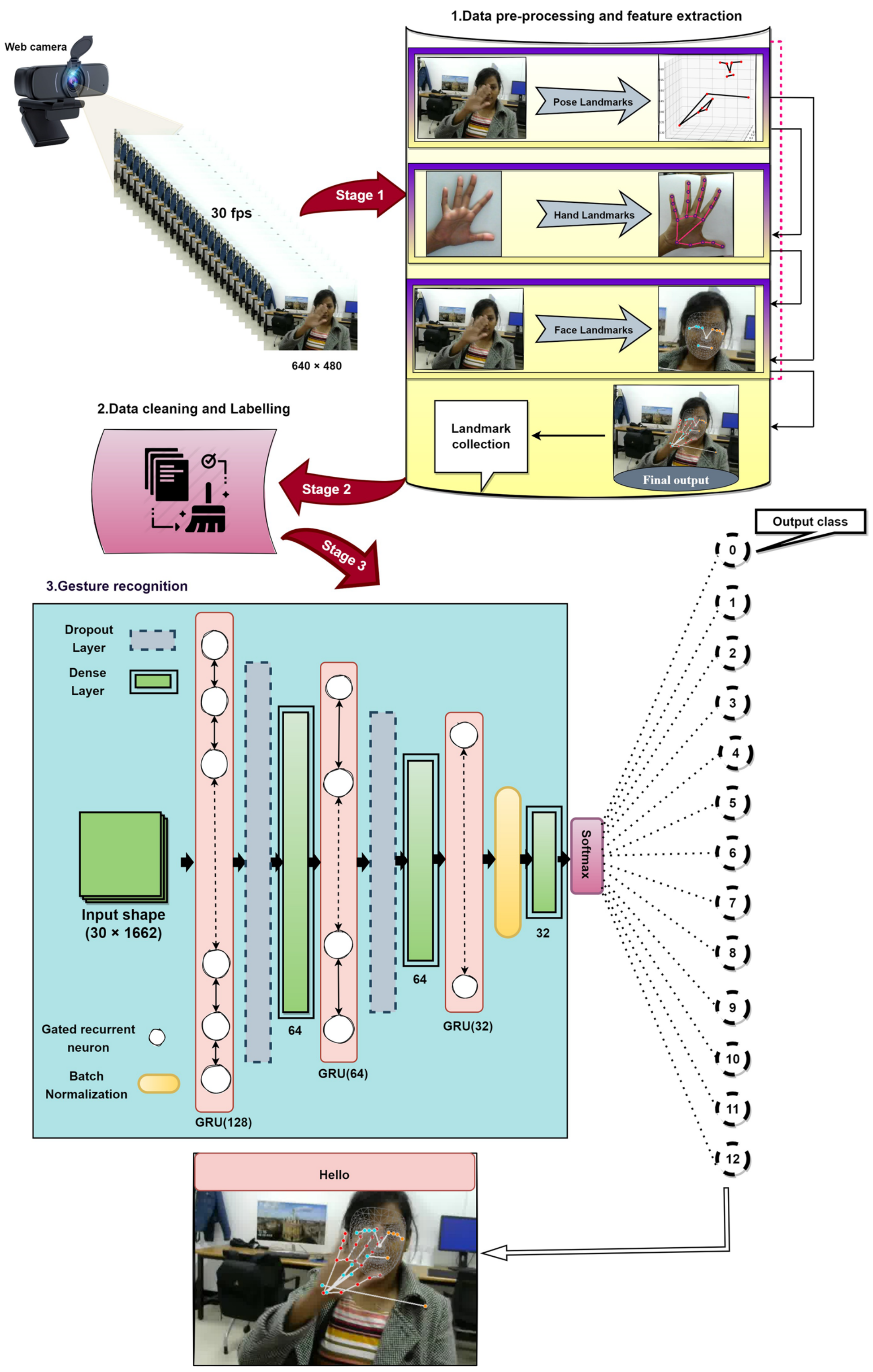

Figure 3. General overview of our proposed architecture.

features[56–58], which occurs when a blurred image is sent to the detector and leads to a null entry into the dataset. Thus, when training occurs with this noisy data, the prediction accuracy is reduced and bias may occur. To fit the obtained data for the next stage of training, testing and validation, labels are created for each class and their corresponding frame sequences are stored.

Stage 3: gesture recognition. *Our proposed MOPGRU model.* The cleaned and labelled data from stage 2, are then passed to stage 3. The major alteration performed in the standard GRU cell is that its update gate is

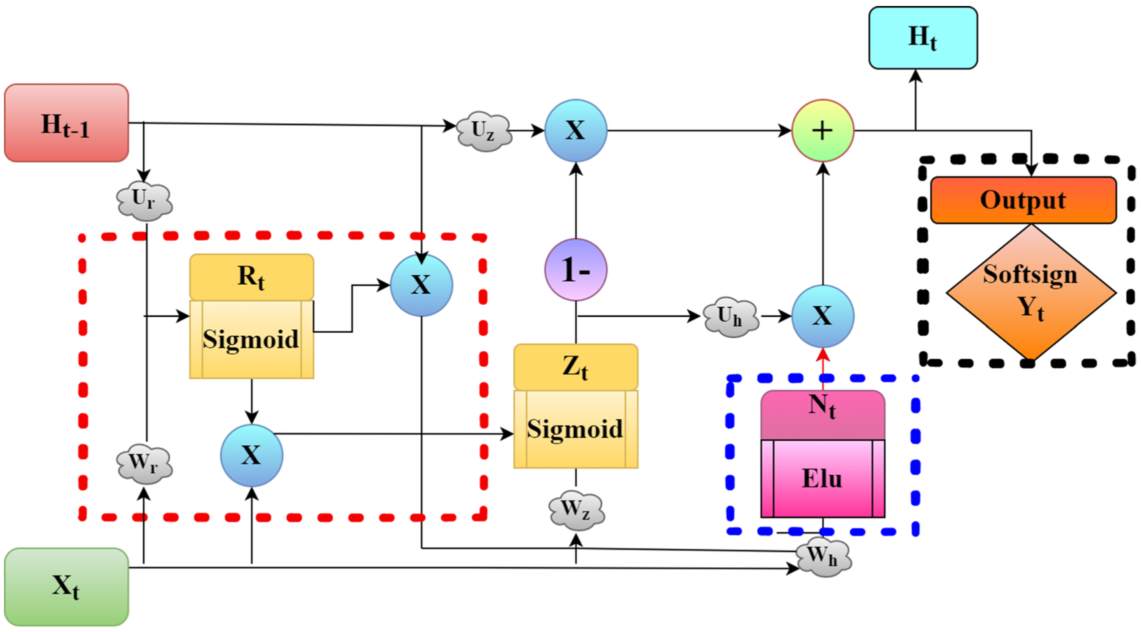

Figure 4. Structure of our proposed GRU cell.

improved, the candidate memory activation function (tanh) in Eq. (3) is substituted by the ELU activation function, and SoftMax activation function in Eq. (5) by Softsign respectively.

Improving the update gate of the standard GRU cell. Considering the problems, such as low learning efficiency, high computational cost, slow convergence rate and incapable of dropping out the unwanted information in one screening with the complex state of time series data of the standard GRU model, we propose a MOPGRU neural network model by improving the update gate; that is, modifying the original update gate input X_t to X_t multiplied by R_t. Thus, the update gate is adjusted by obtaining feedback from the output of the reset gate. The opposite effects caused by unnecessary information are profoundly avoided by refining the present input information X_t by the reset gate attains faster convergence and efficient in learning. Figure 4 shows the neuronal structure of our proposed GRU cell. The red dashed box represents the changes made in the update gate with the reset gate from the standard GRU; the blue dashed box shows the standard hyperbolic tangent activation replaced with ELU activation and the black dashed box shows the SoftMax activation replaced with Softsign activation.

The formula remains unchanged as mentioned in the standard GRU except for Eqs. (2), (3) and (5). The modified formula for the proposed MOPGRU model is as follows:

$$Z_t = \delta((W_z \cdot X_t + U_z \cdot H_{t-1}) * R_t + B_z) \tag{10}$$

Here, the symbols Z_t and R_t in Eq. (10) hold the same meaning as in the standard GRU cell unit, except in Z_t unit structure. The reset gate, R_t, is multiplied by the input vector X_t and then, by the previous time step H_{t-1} in Eq. (10) helped to conceal the state weight, so that the resultant of R_t re-screens the present input X_t by adjusting Z_t to optimize the neuron structure. Our improved GRU cell will not cause any change in computing the derivative of the loss functions as the weights are not changed. Thus, the proposed MOPGRU model makes more sense than the standard GRU, reduces the hidden state and conceals the impoverished gradient to a limited extent. Therefore, our proposed model preserves the information dependency of a longer distance, while producing higher learning efficiency and prediction accuracy.

Incorporating ELU. The other change implemented in the standard GRU cell was replacing the candidate memory tanh activation function with the ELU function, as shown in Fig. 4. This is highlighted with a blue dashed box. Thus, Eq. (3) in the standard GRU cell for calculating the candidate memory N_t changes to the following form:

$$N_t = ELU(W_h \cdot X_t + U_h(H_{t-1} \odot R_t)) + B_h \tag{11}$$

The use of the *tanh* activation function in training feed-forward connections, especially in standard GRU, is ineffective as shown in the performance decline when the network has deeper connections[59]. Additionally, it is computationally expensive and has a vanishing gradient due to its exponential operation. Similarly, the rectified linear unit (ReLU) activation has the dying ReLU problem as its derivative is 0 for negative inputs, meaning that weights are not updated during backpropagation, which leads to zero gradients and dead neurons. Additionally, using it for long-range sequences leads to numerical instability because of its unbounded nature[60].

The ELU activation function given below in Eq. (12) was employed to determine the candidate memory state for the following reasons:

- Using ELU in deep neural networks results in higher classification accuracy and speedy learning[61].
- It does not suffer from vanishing and exploding gradient problems because of its non-saturating characteristics.
- Since ELU is continuous and differentiable, the problem of dying neurons is solved.
- Compared to other activation functions, ELU achieves higher accuracy and faster convergence in less training time and computational complexity since the negative ELU value permits the mean unit to shift toward 0[61].

Mathematically, the ELU is defined as,

$$ELU(x) = \left\{ \begin{array}{ll} x, & if\ x > 0 \\ \alpha(exp(x) - 1), & if\ x \leqslant 0 \end{array} \right\} \tag{12}$$

and its corresponding derivative is defined as,

$$ELU'(x) = \left\{ \begin{array}{ll} 1, & if\ x > 0 \\ ELU(x) + \alpha, & if\ x \leqslant 0 \end{array} \right\} \tag{13}$$

where α is the ELU hyperparameter that controls the value of negative inputs. From Eq. (13), for all positive input values x, the function simply returns the corresponding output Y. However, if the input x is negative, the corresponding output Y will be $exp(x) - 1$. The output of the derivative function moves closer to one. In order to understand the usage of ELU better, let us consider the gradient equation in the BPTT algorithm[62]:

$$\frac{\partial E}{\partial W} = \sum_{k<=t} \frac{\partial E_t}{\partial W} \tag{14}$$

$$= \sum_{k<=t} \frac{\partial E_t}{\partial \hat{Y}_t} \frac{\partial \hat{Y}_t}{\partial H_t} \frac{\partial H_t}{\partial H_k} \frac{\partial^+ H_k}{\partial W} \tag{15}$$

where

$$\frac{\partial H_t}{\partial H_k} = \frac{\partial H_t}{\partial H_{t-1}} \frac{\partial H_{t-1}}{\partial H_{t-2}} \cdots \frac{\partial H_{k+1}}{\partial H_k} \tag{16}$$

Thus the total gradient is calculated using the Eqs. (14), (15) and (16) with the weight matrix W contains distinctive weights for current input and previous hidden state for each gate. Each Jacobian $\frac{\partial H_{k+1}}{\partial H_k}$ is a product of two matrices is a item of two frameworks: the repetitive weight matrix and diagonal matrix composed of the subordinate of non-linearity,ELU, related with the hidden units.In nonappearance of any input, i.e.,$X_t = 0$, and with the choice of starting conditions, the two-norm of each Jacobian in Eq. (14) is indistinguishably one and the error gradients don't develop or decay exponential over time.

Softsign function for output prediction of the GRU cell. The output layers of the neural network use the SoftMax or Sigmoid activation functions for multivariate or binary classification problems, respectively. Softmax activation is specifically used to normalize the outputs and convert the weighted sum values to probabilities that sum to one by exponentiating the features and scale with the sum of the exponents. Because of its exponential nature, it is computationally expensive and time-consuming to train the model. Hence, we incorporated the Softsign activation function as mentioned in Eq. (17) in the output layer of each GRU neurons as shown in Eq. (18), to reduce the time as it finds the quadratic polynomials rather than exponentially. Additionally, as it is zero-centered, the networks learn effectively and the saturation in the network does not occur easily. Like tanh activation , the Softsign ranges from -1 to 1, and is defined as

$$Softsign(x) = \frac{x}{|x| + 1} \tag{17}$$

where |x| is the absolute value of the input point of x. Thus, Eq. (5) in the standard GRU cell for output prediction Y_t changes to the following form:

$$Y_t = Softsign((W_o * H_t) + B_o). \tag{18}$$

With changes made in Eqs. (10), (12) and (18) the MOPGRU model finally displays the recognized gesture result on the screen with its corresponding English text translation, as shown in Fig. 3 from the output of stage 3.

Consent to participate. Informed consent was obtained from the subject for publication of identifying information/images in an online open-access publication.

Experimental results and discussion

Dataset. A real-time dataset of 30 videos for each sign gesture with 30 frames was created, each with a size of 640×480, using a web camera in various directions and different lighting conditions. There are 13 sign gestures in the dataset, as described in Table 1. We separated the collected dataset in the ratio of 70:15:15 to form the corresponding training, testing, and validation datasets. Thus, from the 900 images for each sign gesture, 780

Labels	Sign gestures (words)
0	Fail
1	Friend
2	Good
3	Hello
4	I love you
5	Like
6	Location
7	Meet
8	Phone call
9	Take care
10	Thank you
11	Think
12	You

Table 1. 13 sign gestures and its label.

Network model	MAE	MSE	R^2
Simple RNN	4.10	28.90	− 1.38
LSTM	0.75	4.95	0.59
Standard GRU	0.44	1.38	0.83
BiGRU	0.40	2.50	0.79
BiLSTM	0.85	5.35	0.56
MOPGRU	0.22	1.34	0.88

Table 2. Comparison on MAE, MSE, and R^2 for different models.

images were used for the training set and the remaining 120 images were divided equally, providing 60 images each for testing and validation purposes. Additionally, we evaluated our model on two benchmark datasets: the Word Level American Sign Language (WLASL) dataset[63] and the LSA64 dataset[41]. The WLASL was created for teaching of sign language and hence the data was collected from multiple public resources that includes variety of signing styles and different video backgrounds.The LSA64 dataset contains 3200 videos of 64 isolated sign gestures from the Argentinian sign language which includes verbs and nouns, performed by 10 different people for each word.

Experimental setting. The simulation was conducted using Python 3.7 version on a desktop computer with 32 GB RAM and an Intel Core i7 processor with a frequency of 3.60 GHz, running on Windows 10 Pro with a 64-bit operating system. The input image was captured using a web camera with a resolution of 720pixels/30 fps of RGB images. A sequential model of nine layers were created. Of the nine layers, three are GRU layers, a batch normalization layer, two dropout layers, and three dense layers. The first layer of GRU accepts a sequence of landmark keypoints (1662) extracted from the frames of each video with the time step of 30 as each video contains 30 frame sequences. For the hidden units at the first time-step, the input-hidden vectors and all elements of the initial hidden state were set to 0.The number of hidden units per layer for the model was set to 128, 64 and 32 respectively. The total number of parameters was approximately 450,445.The dropout ratio was fixed at 20% for the hidden and fully connected layers. The training data were set to 100 epochs, with 13 gestures in each batch. To optimize the network, the Adam optimization[64] method was used and was set to 10^{-4} with exponential decay rates of 0.9 and 0.999, respectively. The class scores were calculated using the SoftMax activation function with batch normalized input from the fully connected layer. The number of hidden units per layer for the model was set to 128, 64 and 32 respectively. The total number of parameters was approximately 450,445.

Evaluation metrics. To evaluate the performance of our MOPGRU model, we used the mean squared error (MSE), mean absolute error (MAE), and R-squared or coefficient of determination metrics, as listed in Table 2. MAE denotes the average of the absolute difference between the predicted and actual values in the dataset. It is given by the following formula:

$$MAE = \frac{1}{N}\sum_{i=1}^{N} |y_i - \hat{y}| \tag{19}$$

MSE is the average of the squared difference between the predicted and actual values in the dataset and is calculated by the following formula:

Class label	Simple RNN	LSTM	BiLSTM	Standard GRU	BiGRU	MOPGRU*
0	1	1	1	1	0.5	**1**
1	–	–	–	1	0	**1**
2	1	1	1	1	1	**1**
3	1	1	1	1	1	**0.75**
4	1	0	0	0	1	**1**
5	0	0	0	1	1	**1**
6	0.67	0.67	0.67	1	0	**1**
7	1	0.75	0.75	–	–	**1**
8	1	1	1	1	1	**1**
9	0.5	1	1	1	1	**1**
10	1	1	1	–	–	**1**
11	0	0	0	0	1	**1**
12	1	1	1	0.5	1	**0.92**

Table 3. Precision. Significant values are in [bold].

Class label	Simple RNN	LSTM	BiLSTM	Standard GRU	BiGRU	MOPGRU*
0	0.33	1	1	1	1	**1**
1	–	–	–	0.5	0	**1**
2	1	1	1	1	1	**1**
3	0.5	1	1	1	1	**1**
4	1	0	0	0	1	**0.86**
5	0	0	0	1	1	**1**
6	1	1	1	1	0	**1**
7	1	1	1	–	–	**1**
8	1	1	1	1	1	**1**
9	1	1	1	1	1	**1**
10	1	1	1	–	–	**0.7**
11	0	0	0	0	0.5	**1**
12	1	0.5	0.5	1	1	**1**

Table 4. Recall. Significant values are in [bold].

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y})^2 \tag{20}$$

R^2 score indicates how well the model fits the given dataset. It indicates how close the predicted value is to the actual data values. The value lies between 0.0 and 1.0, where 0.0 indicates the worst fit and 1.0 indicates the perfect fit. It was calculated using the following formula:

$$R^2 = 1 - \frac{\sum(y_i - \hat{y})^2}{\sum(y_i - \bar{y})^2} \tag{21}$$

where $\hat{y}$ represents the predicted value of y, and $\bar{y}$ denotes the mean value of y. Error calculations were performed from Eqs. (19), (20), and (21). From Table 2, we see that the MAE and MSE values are higher for simple RNN, LSTM, Standard GRU, bidirectional GRU (BiGRU), and deep bidirectional LSTM (BiLSTM), which means that the average residual and variance of the residual are high. Previously BiLSTM showed good performance in action recognition[65], whereas training LSTM and BiLSTM models were not beneficial for our datasets because we had limited data for sequence prediction. Furthermore, we attempted to produce comparatively low prediction errors using our proposed MOPGRU model.

Quantitative analysis. The classification metrics were used to analyze the quality of prediction for each sign gesture, as shown in Tables 3, 4, and 5, respectively. These qualities include precision, recall, and F1-score[66–68], which are calculated using four values: true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN). We conducted experiments using different models as mentioned in Table 2 by inserting each model in the place of our proposed MOPGRU layers in the architecture presented in the paper. The number of correctly predicted data points is called accuracy, and ideally, it must be close to one. The accuracy of the SLR system is calculated as follows:

Class label	Simple RNN	LSTM	BiLSTM	Standard GRU	BiGRU	MOPGRU*
0	0.5	1	1	1	0.67	**1**
1	–	–	–	0.67	0	**1**
2	1	1	1	1	1	**1**
3	0.67	1	1	1	1	**0.86**
4	1	0	0	0	1	**0.92**
5	0	0	0	1	1	**1**
6	0.8	0.8	0.8	1	0	**1**
7	1	0.86	0.86	–	–	**1**
8	1	1	1	1	1	**1**
9	0.67	1	1	1	1	**1**
10	1	1	1	–	–	**0.82**
11	0	0	0	0	0.67	**1**
12	1	0.67	0.67	0.67	1	**0.96**

Table 5. F1-Score. Significant values are in [bold].

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{22}$$

Precision is the number of positive predictions divided by the total number of positive class values predicted, which is called the positive predicted value. High cost of false positive makes precision an important measure to determine. Like accuracy, this must also be close to one. The precision value is calculated as follows:

$$Precision = \frac{TP}{TP + FP} \tag{23}$$

A recall is the fraction of positive events predicted correctly by the model and high cost of false negative makes recall an important measure to determine. It is calculated as follows:

$$Recall = \frac{TP}{TP + FN} \tag{24}$$

The F1-score or the F-measure is the harmonic mean of precision, and recall means it conveys a balance between recall and precision. When there is perfect precision and recall, the F1 score reaches its best value at 1 and can be calculated using the following formula:

$$F1 = \frac{2TP}{2TP + FP + FN} \tag{25}$$

Based on the calculations given in Eqs. (22), (23), (24) and (25), we obtain a classification report as shown in Tables 3, 4 and 5. From the classification metrics, the precision, recall, and F1-score of MOPGRU model are almost 1 except for two values in both precision and recall, and four values in F1-score, which are still ideally close to 1. This shows that our model effectively learned data on complete training. In Contrary, other models such as LSTM, Simple RNN, Standard GRU, BiLSTM and BiGRU resulted in good accuracy, but their learning efficiency is not good. As a result, these models did not perform well as they contain many zero metric scores for precision, recall, and F1-score.

Considering the R^2 values, our proposed MOPGRU outperforms the other three models with the highest score of 0.88, which is closer to 1, indicating that the model has a good fit. However, a negative score in the Simple RNN indicates that the model has the worst fit. Figure 5a and b show the training accuracy and loss of different models and Fig. 6a and b show the validation accuracy and loss of different models used to classify 13 individual sign gestures from the dataset. From Fig. 5, we can conclude that the training is unstable due to the fluctuation between the accuracy and epochs, as well as with loss and epochs. However, in the case of MOPGRU, the training is smooth and faster with efficient data learning. Therefore, our proposed model produces higher performance with minimum loss compared with other models, such as RNN, LSTM, standard GRU, BiGRU, and BiLSTM, respectively. By comparing the models with respect to both the test accuracy and loss, as shown in Figs. 7 and 8, we conclude that our proposed MOPGRU model achieved the highest test accuracy of 95% compared to other models such as LSTM with 85%, Simple RNN with 80% standard GRU with 85% accuracy, BiGRU with 90%, and BiLSTM with 85% and minimum loss of 0.21 compared to others.

Comparative analysis. To access the efficiency and performance of our proposed MOPGRU model, we conducted experiments on the two different benchmark datasets LSA64 and WLASL100 respectively and compared our results with different existing state-of-the art models like I3D, Pose-based temporal graph convolutional network (Pose-TGCN) and Pose-based gated recurrent unit (Pose-GRU). We train the model with the same parameters mentioned in[63] and achieved high recognition accuracy than the existing method. As compared to other models used in the experiment, the learning efficiency and the convergence speed of our pro-

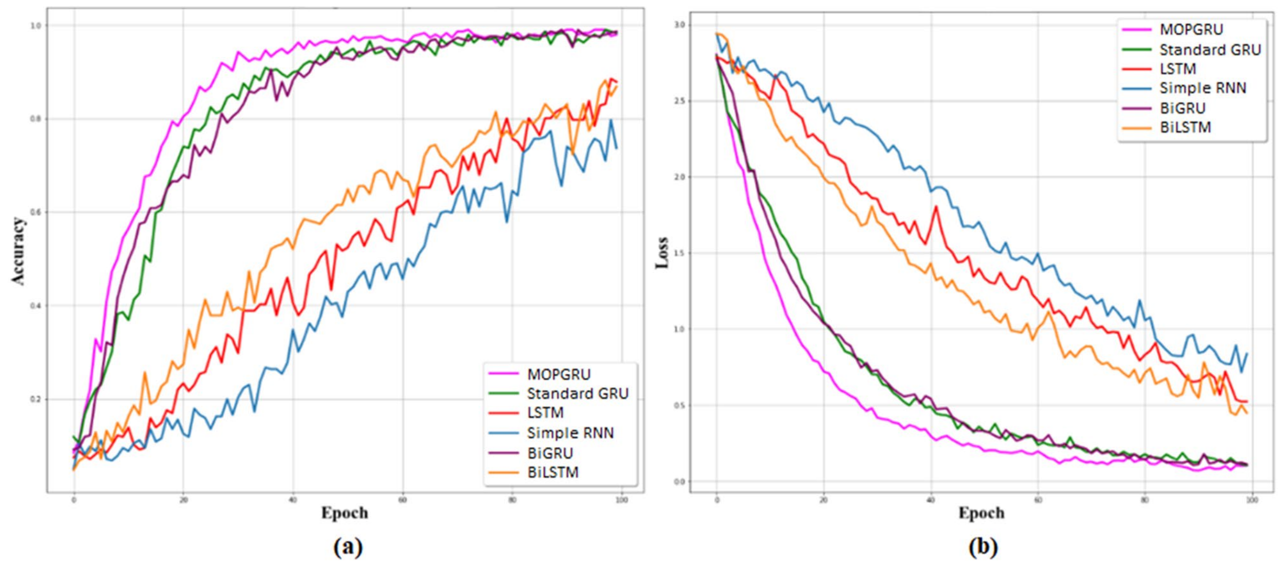

Figure 5. Learning graph with training accuracy and loss for 100 epochs of different models used.

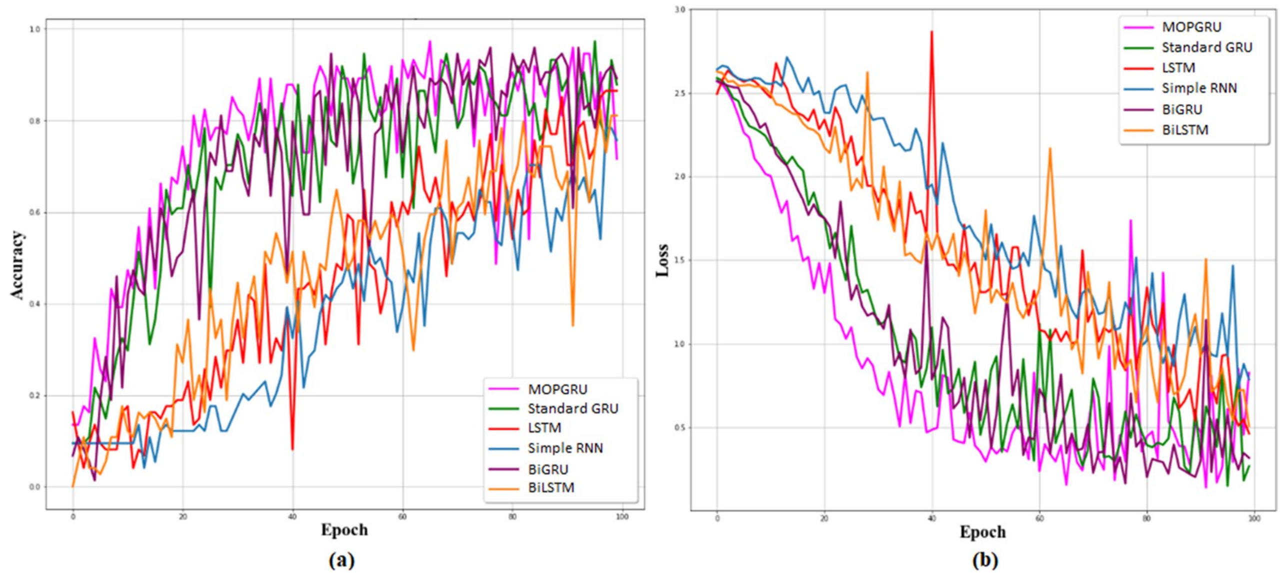

Figure 6. Learning graph with validation accuracy and loss for 100 epochs of different models used.

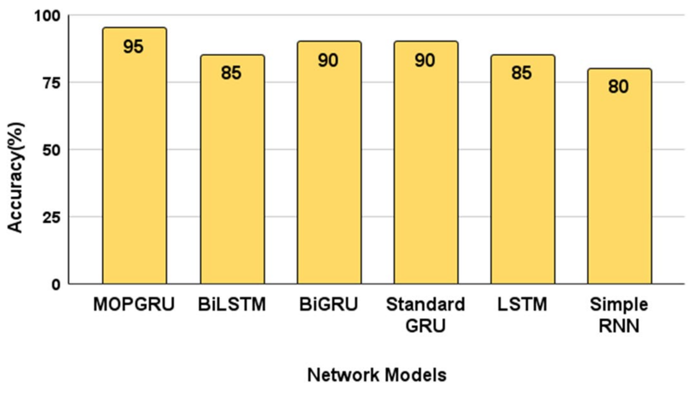

Figure 7. Model comparison in terms of test accuracy in percent (%).

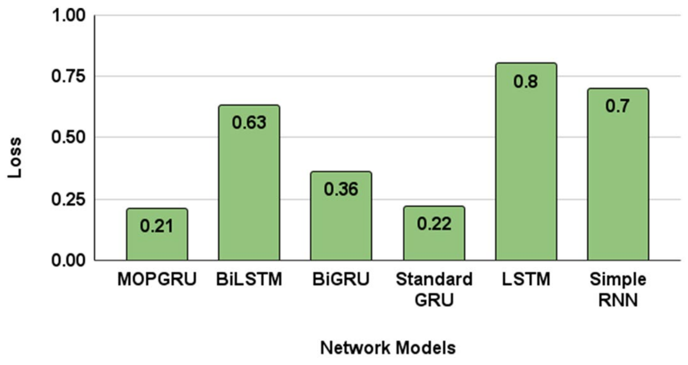

Figure 8. Model comparison in terms of test loss.

Model	Dataset	Accuracy (%)
Pose-TGCN	WLASL100	55.43
Pose-GRU	WLASL100	46.51
MOPGRU	WLASL100	**63.18**
I3D	LSA64	98.91
MOPGRU	LSA64	**99.92**

Table 6. Recognition accuracy achieved by several models on the LSA64 and WLASL100 dataset. Significant values are in [bold].

posed MOPGRU was also high with the use of a smaller number of parameters and the activation function during training. Datasets which are existing have their own properties to deal with the isolated word level sign recognition task. However, they fail to capture the complexities of the task due to inadequate amount of instance and signers. Thus we evaluated our model on two benchmark datasets that have adequate amount of instance and signers. The recognition accuracy on both the datasets are presented in Table 6.

Conclusion

In this study, we proposed a MOPGRU model for ISL recognition. Specifically, we modified the update gate of the standard GRU cell by multiplying its output by the reset gate. With our improved update gate mechanism, the output of the reset gate re-screens the information and removes the unwanted information in the data, thereby giving more attention to the important information. We cost-effectively implemented the model, which resulted in improved learning efficiency, prediction accuracy, and information processing capability of the standard GRU neural network. Furthermore, experimental results showed that compared to simple RNN, LSTM, standard GRU, BiGRU, and BiLSTM prediction models, MAE and MSE values of our proposed GRU neural network model were very low with high R-squared values. Therefore, our proposed MOPGRU captured the full information dependency in time series data with a high prediction accuracy of an average of 95% and a faster convergence speed. Although our model performs well in terms of accuracy and learning efficiency, this study was conducted with a limited dataset. Thus, for future work, we aim to improve our SLR system by expanding the dataset with more vocabulary to predict continuous sign language sentences.

Data availibility

The datasets generated during and/or analysed during the current study are available from the corresponding author on reasonable request.

Received: 28 February 2022; Accepted: 4 July 2022
Published online: 13 July 2022

References

1. Jain, R. K. & Rathi, S. K. A review paper on sign language recognition using machine learning techniques. In *Emerging Trends in Data Driven Computing and Communications* (eds Mathur, R. *et al.*) (Springer, 2021).
2. Aloysius, N., Geetha, M. & Nedungadi, P. Incorporating relative position information in transformer-based sign language recognition and translation. *IEEE Access* **9**, 145929–145942 (2021).
3. Li, D., Rodríguez, C., Yu, X. & Li, H. Word-level deep sign language recognition from video: A new large-scale dataset and methods comparison. arXiv:1910.11006v2 (2019).
4. Kadhim, R. A. & Khamees, M. A real-time American sign language recognition system using convolutional neural network for real datasets. *TEM J.* **9**(3), 937–943 (2020).

5. Wadhawan, A. & Kumar, P. Deep learning-based sign language recognition system for static signs. *Neural Comput. Appl.* **32**, 7957–7968 (2020).
6. Zafrulla, Z., Brashear, H., Starner, T., Hamilton, H. & Presti, P. American sign language recognition with the kinect. In *Proceedings of the 13th International Conference on Multimodal Interfaces (ICMI '11)* 279–286 (Association for Computing Machinery, 2011). https://doi.org/10.1145/2070481.2070532.
7. Kumar, P., Gauba, H., Roy, P. P. & Dogra, D. P. Coupled HMM-based multi-sensor data fusion for sign language recognition. *Pattern Recognit. Lett.* **86**(C), 1–8 (2017).
8. Kumar, P., Gauba, H., Roy, P. P. & Dogra, D. P. A multimodal framework for sensor based sign language recognition. *Neurocomputing* **259**, 21–38. https://doi.org/10.1016/j.neucom.2016.08.132 (2017).
9. Elakkiya, R. & Selvamani, K. Subunit sign modeling framework for continuous sign language recognition. *Comput. Electr. Eng.* **74**, 379–390. https://doi.org/10.1016/j.compeleceng (2019).
10. Gadekallu, T. R. *et al.* Hand gesture classification using a novel CNN-crow search algorithm. *Complex Intell. Syst.* **7**, 1855–1868 (2021).
11. Ibrahim, N. B., Zayed, H. & Selim, M. Advances, challenges and opportunities in continuous sign language recognition. *J. Eng. Appl. Sci.* **15**(5), 1205–1227 (2019).
12. Koller, O. Quantitative survey of the state of the art in sign language recognition. *arXiv* (2020).
13. Mittal, A., Kumar, P., Roy, P. P., Balasubramanian, R. & Chaudhuri, B. B. A modified LSTM model for continuous sign language recognition using leap motion. *IEEE Sens. J.* **19**, 7056–7063. https://doi.org/10.1109/JSEN.2019.2909837 (2019).
14. Kanisha, B. *et al.* Smart communication using tri-spectral sign recognition for hearing-impaired people. *J. Supercomput.* **78**, 2651–2664 (2022).
15. Sun, Z. A survey on dynamic sign language recognition. In *Advances in Computer, Communication and Computational Sciences* Vol. 1158 (eds Bhatia, S. K. *et al.*) (Springer, 2021).
16. Rakesh, S., Bharadhwaj, A. & Sree, H. E. Sign language recognition using convolutional neural network. In *Innovative Data Communication Technologies and Application* Vol. 59 (eds Raj, J. S. *et al.*) (Springer, 2021).
17. Kiran, Kumar E., Kishore, P. V. V., Sastry, A. S. C. S. & Anil, Kumar D. 3D motion capture for Indian sign language recognition (SLR). In *Smart Computing and Informatics* Vol. 78 (eds Satapathy, S. *et al.*) (Springer, 2018).
18. Itkarkar Rajeshri, R., Nandi, A. K. V. & Mungurwadi, V. B. Indian sign language recognition using combined feature extraction. In *Advances in Medical Physics and Healthcare Engineering* (eds Mukherjee, M. *et al.*) (Springer, 2021).
19. Starner, T., Weaver, J. & Pentland, A. Real-time American sign language recognition using desk and wearable computer based video. *IEEE Trans. Pattern Anal. Mach. Intell.* **20**, 1371–1375. https://doi.org/10.1109/34.735811 (1999).
20. Vogler, C. & Metaxas, D. N. Adapting hidden Markov models for ASL recognition by using three-dimensional computer vision methods. In *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics* Vol. 1 (1970). https://doi.org/10.1109/ICSMC.1997.625741.
21. Shukor, A. Z. *et al.* A new data glove approach for Malaysian sign language detection. *Procedia Comput. Sci.* **76**, 60–67. https://doi.org/10.1016/j.procs.2015.12.276 (2015).
22. Almeida, S., Guimaraes, F. G. & Ramirez, J. A. Feature extraction in Brazilian sign language recognition based on phonological structure and using RGB-D sensors. *Expert Syst. Appl.* **41**(16), 7259–7271. https://doi.org/10.1016/j.eswa.2014.05.024 (2014).
23. Patil, A., Kulkarni, A., Yesane, H., Sadani, M. & Satav, P. Literature survey: Sign language recognition using gesture recognition and natural language processing. In *Data Management, Analytics and Innovation* Vol. 70 (eds Sharma, N. *et al.*) (Springer, 2021).
24. Hurroo, M. & Elham, M. Sign language recognition system using convolutional neural network and computer vision. *Int. J. Eng. Res. Technol. (IJERT)* **9**(12), 59–64 (2020).
25. Rastgoo, R., Kiani, K. & Escalera, S. Hand sign language recognition using multi-view hand skeleton. *Expert Syst. Appl.* **150**, 113336. https://doi.org/10.1016/j.eswa.2020.113336 (2020).
26. Lee, C. K. M. *et al.* American sign language recognition and training method with recurrent neural network. *Expert Syst. Appl.* **167**(October), 114403. https://doi.org/10.1016/j.eswa.2020.114403 (2021).
27. Chen, R.-C., Dewi, C., Huang, S.-W. & Caraka, R. E. Selecting critical features for data classification based on machine learning methods. *J. Big Data* **7**, 52. https://doi.org/10.1186/s40537-020-00327-4 (2020).
28. Gupta, R. & Kumar, A. Indian sign language recognition using wearable sensors and multi-label classification. *Comput. Electr. Eng.* **90**(December), 106898. https://doi.org/10.1016/j.compeleceng.2020.106898 (2020).
29. Grishchenko, I. & Bazarevsky, V. Mediapipe holistic. Retrieved from https://ai.googleblog.com/2020/2012 20/ (2020).
30. Naglot, D. & Kulkarni, M. Recognition using the leap motion controller. In *International Conference on Inventive Computation Technologies (ICICT)* Vol. 2, 1–6 (2016). https://doi.org/10.1109/INVENTIVE.2016.7830097.
31. Bhagat, N. K., Vishnusai, Y. & Rathna, G. N. Indian sign language gesture recognition using image processing and deep learning. In *2019 Digital Image Computing: Techniques and Applications (DICTA)* (2019). https://doi.org/10.1109/DICTA47822.2019.8945850
32. Raghuveera Tripuraribhatla, R., Deepthi, R., Mangalashri, R. & Akshaya, R. A depth-based Indian Sign language recognition using Microsoft Kinect. *Sadhana Acad. Proc. Eng. Sci.* **45**(1), 1–13. https://doi.org/10.1007/s12046-019-1250-6 (2020).
33. Neethu, P. S., Ramadass, S. & Sathish, D. An efficient method for human hand gesture detection and recognition using deep learning convolutional neural networks. *Soft Comput.* **24**(20), 15239–15248. https://doi.org/10.1007/s00500-020-04860-5 (2020).
34. Salem, N., Alharbi, S., Khezendar, R. & Alshami, H. Real-time glove and android application for visual and audible Arabic sign language translation. *Procedia Comput. Sci.* **163**, 450–459. https://doi.org/10.1016/j.procs.2019.12.128 (2019).
35. Rastgoo, R., Kiani, K. & Escalera, S. Real-time isolated hand sign language recognition using deep networks and SVD. *J. Ambient Intell. Humaniz. Comput.*https://doi.org/10.1007/s12652-021-02920-8 *(2021).*
36. Rastgoo, R., Kiani, K. & Escalera, S. Hand pose aware multimodal isolated sign language recognition. *Multimed Tools Appl.* **80**, 127–163. https://doi.org/10.1007/s11042-020-09700-0 (2021).
37. Rastgoo, R., Kiani, K. & Escalera, S. Video-based isolated hand sign language recognition using a deep cascaded model. *Multimedia Tools Appl.* **79**(31–32), 22965–22987. https://doi.org/10.1007/s11042-020-09048-5 (2020).
38. Al-Hammadi, M. *et al.* Hand gesture recognition for sign language using 3DCNN. *IEEE Access* **8**, 79491–79509. https://doi.org/10.1109/ACCESS.2020.2990434 (2020).
39. Chen, C., Liu, L., Wan, S., Hui, X. & Pei, Q. Data dissemination for industry 4.0 applications in internet of vehicles based on short-term traffic prediction. *ACM Trans. Internet Technol.* **22**, 1–18. https://doi.org/10.1145/3430505 (2022).
40. Carreira, J. & Zisserman, A. (2017) Quo vadis action recognition? A new model and the kinetics dataset. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* 6299–6308 (2017).
41. Quiroga, F., Ronchetti, F., Estrebou, C. A., Lanzarini, L. C. & Rosete, A. Lsa64: An argentinian sign language dataset. In *XXII Congreso Argentino de Ciencias de la Computación* 794–803 (CACIC, 2016).
42. Ojha, A., Pandey, A., Maurya, S., Thakur, A. & Dayananda, P. Sign language to text and speech translation in real time using convolutional neural network. *Int. J. Eng. Res. Technol. (IJERT)* **8**(15), 191–196 (2020).
43. Koller, O., Ney, H. & Bowden, R. Deep hand: How to train a CNN on 1 million hand images when your data is continuous and weakly labelled. In *IEEE International Conference on Computer Vision and Pattern Recognition* Vol. 2016-Decem, 3793–3802 (2016). https://doi.org/10.1109/CVPR.2016.412

44. Mocialov, B., Turner, G. H., Lohan, K. S. & Hastie, H. Towards continuous sign language recognition with deep learning. In *Proceedings of the Workshop on the Creating Meaning With Robot Assistants: The Gap Left by Smart Devices* (2017).
45. Molchanov, P., Yang, X., Gupta, S., Kim, K., Tyree, S. & Kautz, J. Online detection and classification of dynamic hand gestures with recurrent 3D convolutional neural networks. In *IEEE International Conference on Computer Vision and Pattern Recognition* vol. 2016-December, 4207–4215 (2016).
46. Elakkiya, R. & Selvamani, K. Enhanced dynamic programming approach for subunit modelling to handle segmentation and recognition ambiguities in sign language. *J. Parallel Distrib. Comput.* **117**, 246–255. https://doi.org/10.1016/j.jpdc.2017.07.001 (2018).
47. Cheok, M. J., Omar, Z. & Jaward, M. H. A review of hand gesture and sign language recognition techniques. *Int. J. Mach. Learn. Cybern.* **10**(1), 131–153. https://doi.org/10.1007/s13042-017-0705-5 (2019).
48. Nai, W., Liu, Y., Rempel, D. & Wang, Y. Fast hand posture classification using depth features extracted from random line segments. *Pattern Recognit.* **65**(November), 1–10. https://doi.org/10.1016/j.patcog.2016.11.022 (2017).
49. Elakkiya, R. Machine learning based sign language recognition: a review and its research frontier. *J. Ambient Intell. Humaniz. Comput.*https://doi.org/10.1007/s12652-020-02396-y *(2021).*
50. Adithya, V., Vinod, P. R. & Gopalakrishnan, U. Artificial neural network based method for Indian sign language recognition. In *2013 IEEE Conference on Information and Communication Technologies (ICT)* 1080–1085 (2013). https://doi.org/10.1109/CICT.2013.6558259.
51. Meng, X. J., Qiu, S., Wan, S., Cheng, K. & Cui, L. A motor imagery EEG signal classification algorithm based on recurrence plot convolution neural network. *Pattern Recognit. Lett.* **134146**, 134–141. https://doi.org/10.1016/j.patrec.2021.03.023 (2021). ISSN 0167-8655.
52. Xiao, L., Fan, C., Ouyang, H., Abate, A. F. & Wan, S. Adaptive trapezoid region intercept histogram based Otsu method for brain MR image segmentation. *J. Ambient Intell. Hum. Comput.* **13**, 2161–2176. https://doi.org/10.1007/s12652-021-02976-6 (2022).
53. Lyu, Y. & Huang, X. Road segmentation using CNN with GRU. *Comput. Vis. Pattern Recognit.*arXiv:1804.05164 *(2018).*
54. Cho, K., Van Merriënboer, B., Bahdanau, D. & Bengio, Y. On the properties of neural machine translation: Encoder–decoder approaches. *Comput. Lang.*arxiv:1409.1259 *(2014).*
55. Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H. & Bengio, Y. Learning phrase representations using RNN encoder–decoder for statistical machine translation. *Comput. Lang.* Retrieved from arxiv:1406.1078 (2014).
56. Olimov, B. *et al.* Weight initialization based-rectified linear unit activation function to improve the performance of a convolutional neural network model. *Pract. Exp. Concurr. Comput.*https://doi.org/10.1002/cpe.6143 *(2020).*
57. Olimov, B., Kim, J. & Paul, A. Deep clean before training network: Training deep convolutional neural networks with extremely noisy labels. *IEEE Access* **8**, 220482–220495. https://doi.org/10.1109/ACCESS.2020.3041873 (2020).
58. Olimov, B., Kim, J. & Paul, A. REF-Net: Robust, efficient, and fast network for semantic segmentation applications using devices with limited computational resources. *IEEE Access* **9**, 15084–15098. https://doi.org/10.1109/ACCESS.2021.3052791 (2021).
59. Gulcehre, C., Moczulski, M., Denil, M. & Bengio, Y. Noisy activation functions. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning, (ICML'16)* Vol. 48. JMLR.org, 3059–3068 (2016).
60. Ravanelli, M., Brakel, P., Omologo, M. & Bengio, Y. Light gated recurrent units for speech recognition. *IEEE Trans. Emerg. Top. Comput.* **2**(2), 92–102. https://doi.org/10.1109/TETCI.2017.2762739 (2018).
61. Clevert, D.A., Unterthiner, T. & Hochreiter, S. Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUS). arXiv:1511.07289v5 (2015).
62. Pascanu, R., Mikolov, T. & Bengio, Y. On the difficulty of training recurrent neural networks. In *Proceedings of the 30th International Conference on International Conference on Machine Learning, (ICML'13)* Vol. 28. JMLR.org, III-1310–III-1318 (2013).
63. Li, D., Rodriguez, C., Yu, X. & Li, H. Word-level deep sign language recognition from video: A new large-scale dataset and methods comparison. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision* 1459–1469 (2020).
64. Kingma, D. P. & Ba, J. Adam: A method for stochastic optimization. arxiv:1412.6980 (2014).
65. Ullah, A., Ahmad, J., Muhammad, K., Sajjad, M. & Baik, S. W. Action recognition in video sequences using deep bi-directional LSTM with CNN features. *IEEE Access* **6**, 1155–1166. https://doi.org/10.1109/ACCESS.2017.2778011 (2017).
66. Olimov, B. *et al.* FU-Net: Fast biomedical image segmentation model based on bottleneck convolution layers. *Multimedia Syst.* **27**, 1–14. https://doi.org/10.1007/s00530-020-00726-w (2021).
67. Olimov, B., Koh, S.-J. & Kim, J. AEDCN-Net: Accurate and efficient deep convolutional neural network model for medical image segmentation. *IEEE Access*https://doi.org/10.1109/ACCESS.2021.3128607 *(2021).*
68. Olimov, B., Kim, J., Paul, A. & Subramanian, B. An efficient deep convolutional neural network for semantic segmentation. In *8th International Conference on Orange Technology (ICOT)* 1–9 (2020). https://doi.org/10.1109/ICOT51877.2020.9468748.

Acknowledgements
This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (2021R1I1A3043970). This study was also supported by the BK21 FOUR project (AI-driven Convergence Software Education Research Program) funded by the Ministry of Education, School of Computer Science and Engineering, Kyungpook National University, Korea (4199990214394).

Author contributions
B.S. conceived the idea and wrote the main manuscript. B.O. and S.M.N. contributed to the data analysis and edited the manuscript. S.K., K.-H.P. and J.K. supervised the work.

Competing interest
The authors declare no competing interests.

Additional information
Correspondence and requests for materials should be addressed to J.K.

Reprints and permissions information is available at www.nature.com/reprints.