Definitions/applsci-logo-eps-converted-to.pdf

Article

BenSignNet: Bengali Sign Language Alphabet Recognition Using Concatenated Segmentation and Convolutional Neural Network

Abu Saleh Musa Miah [1], Jungpil Shin [1,*], Md. Al Mehedi Hasan [1] and Md Abdur Rahim [2]

[1] School of Computer Science and Engineering, The University of Aizu, Aizuwakamatsu 965-8580, Fukushima, Japan; d8231105@u-aizu.ac.jp (A.S.M.M.); mehedi@u-aizu.ac.jp (M.A.M.H.)
[2] Department of Computer Science and Engineering, Pabna University of Science and Technology; rahim@pust.ac.bd
* Correspondence: jpshin@u-aizu.ac.jp

Abstract: Sign language recognition is one of the most challenging applications in machine learning and human-computer interaction. Many researchers have developed classification models for different sign languages such as English, Arabic, Japanese, and Bengali; however, no significant research has been done on the general-shape performance for different datasets. Most research work has achieved satisfactory performance with a small dataset. These models may fail to replicate the same performance for evaluating different and larger datasets. In this context, this paper proposes a novel method for recognizing Bengali sign language(BSL) alphabets to overcome the issue of generalization. The proposed method has been evaluated with three benchmark datasets such as '38 BdSL', 'KU-BdSL', and 'Ishara-Lipi'. Here, three steps are followed to achieve the goal: segmentation, augmentation, and Convolutional neural network (CNN) based classification. Firstly, a concatenated segmentation approach with YCbCr, HSV and watershed algorithm was designed to accurately identify gesture signs. Secondly, seven image augmentation techniques are selected to increase the training data size without changing the semantic meaning. Finally, the CNN-based model called BenSignNet was applied to extract the features and classify purposes. The performance accuracy of the model achieved **94%**, **99.60%**, and **99.60%** for the BdSL Alphabet, KU-BdSL, and Ishara-Lipi datasets, respectively. Experimental findings confirmed that our proposed method achieved a higher recognition rate than the conventional ones and accomplished a generalization property in all datasets for the BSL domain.

Keywords: Bengali sign language (BSL); Convolutional neural network (CNN); 38-BdSL; Ishara-Lipi; KU-BdSL; concatenated segmentation; Luminance blue red (YCbCr); Hue saturation value (HSV)

Citation: Miah A.S.M.; Shin, J.; Hasan, M.A.M.; Rahim, A. BenSignNet: Bengali Sign Language Alphabet Recognition Using Concatenated Segmentation and Convolutional Neural Network. *Appl. Sci.* **2022**, *1*, 0. https://doi.org/

Academic Editor: Firstname Lastname

Received:
Accepted:
Published:

1. Introduction

The deaf and hard of hearing (DHH) people can do many things in their daily lives, excluding communication. Although, communication is inevitable for passing the message and expressing one's ideas, thoughts, and social identity to others. DHH community used sign language to communicate with others instead of the common language. This is also the primary mode of communication for the DHH community in Bangladesh, although it is difficult for general people to communicate with deaf and mute individuals. Sign language is a structural form of hand gestures involving signs of different body parts, such as fingers, hands, arms, head, body, and facial expressions, that are used as a communication system to help the deaf and speech-impaired community for daily interaction [1]. There are over 70 million deaf and hard of hearing people worldwide [2], with nearly 3 million deaf people in Bangladesh [3]. Moreover, due to a lack of social awareness, the deaf and hard of hearing group has communication difficulties on their own. For this, the primary and everyday activities such as education [4,5], medical services, employment, and socializing have been challenged. In the given scenario, a sign language interpreter

is required for communication between the general and deaf communities. However, an experienced translator may not always be accessible, and paying reasonable costs in such cases may be a major concern. As a solution, automatic sign language recognition can play an essential role in bridging the fundamental and social gap between the deaf and healthy communities. Nowadays, researchers are focused on two domains to develop automated sign language systems [6], such as sensor-based [7] and vision-based approaches [8]. Most BSL researchers focus on vision-based approaches with machine learning and the deep learning technique [9,10], which capture movements statically or dynamically utilizing one-handed and two-handed techniques.

There are a few systems has been developed for BSL recognition. Moreover, the benchmark dataset on BSL is inadequate, and there are few benchmark datasets that are freely available to evaluate the existing model and accurately train the deep learning model. Among them, 38 BdSL datasets introduced by Rafi et al., which contained 12,160 samples, produced performance accuracy using deep learning is **89.60%**, which is very low for implementing a BSL recognition system [11]. KU-BdSL and Ishara-Lipi also considered benchmark datasets of BSL that are publicly available [12,13]. Most of the BSL researchers are conducting their research based on their own datasets [9,14,15]. The authors achieved a satisfactory level of accuracy on the small author-created datasets; however, these systems may fail to produce a good performance for other benchmark datasets. On the other hand, a diverse variety of sign gesture, complicated environment, partial occultation, redundant back-ground, and viewpoint of light variation are frequent problems for achieving high performance in BSL recognition. By considering these challenges, a generalized system for BSL recognition is inevitable for implementing a real-time BSL recognition system. This study aims to investigate the current state of the art system for BSL recognition and deploy a generalized BSL recognition system to increase the performance for the real-world scenarios where the data come from different diverse. The main contribution of this paper is described below:

- We proposed concatenated segmentation techniques to solve light illumination, uncontrolled environment and background noise. Segmentation techniques consist of YCbCr, HSV, morphology and watershed algorithms.
- We used seven augmentation approaches to generate diverse sign images such as rotated, translated, scaled, flipped from the input image in order to enlarge the dataset, deal with inefficient deep learning model training and keep the model image diversity invariant.
- Finally, we developed a modified robust CNN architecture after adjusting hyperparameters called BenSignNet to increase the generalization property of the system. This makes its image diversity invariant and produces a good performance for diverse BSL datasets such as 38 BdSL, KU-BdSL, and Ishara-Lipi datasets. Based on our knowledge, the proposed BenSignNet is more effective and efficient than the previously reported BSL system. After that, proposed model could be used for rapidly detecting BSL for Bengali DHH community.

We organized the presented work as follows, Section 2 summarize the existing research work and problems which are focused in the presented work, Section 3 describes the three benchmark dataset of BSL and Section 4 discussed data preprocessing and experimental details the proposed work. Section 5 shows the result obtained from the experiment with a different dataset and discussion. In Section 6 conclude the paper including some future work.

2. Related Work

With the deep learning (DL) successful applications in the image classification [16–18] and natural language processing (NLP) field [19–21], it also achieved considerable progress in the DL-based sign language recognition task. In Section 2.1, we described an overview of the work related to BSL. Section 2.2 provide an overview of the research work on other sign languages.

2.1. Literary Review on Bengali Sign Language (BSL)

BSL is the modified form of American, British, Australian, and Indian sign language. Although many researchers have been working to develop a BSL recognizer to help deaf and hard of hearing people in the Bengali community, it is not yet explored as much as needed. Kaushik et al. applied a cross-correlation for two-handed sign language recognition for Bengali characters in 2012 [22]. They used 80 images for ten BSL datasets classes to evaluate their model and achieved 96% accuracy. In the same way, an ensemble neural network was applied in [23], feature-based cascaded classifiers like Haar [24], contour analysis [25], support vector machine(SVM) [26].

Nowadays, CNN based feature extraction and classification approaches are effective for sign language classification due to increasing the size of the dataset, and most of the research work for BSL have developed using this. Yasir et al. in 2017 proposed a CNN based on a Leap motion controller (LMC) to track the hand motion of the signer. They operated this with a limited dataset for 14 classes of BSL, and their error rate is 3% [27]. Haque et al. in 2018 proposed a faster regional-based CNN to detect real-time BSL with ten classes dataset [28]. Authors successfully trained the system on the 1700 images and achieved **98.2%** accuracy for recognizing 10 signs. However, the model's performance has not been confirmed because the author collected a small size dataset that could raise an underfitting problem in their model. To solve the BSL dataset problem, Islam et al. in 2018 built a new 36 class BSL dataset with 1800 samples, and the name of the dataset is Ishara-Lipi [29]. Islam et al. applied a CNN model to recognize the Ishara-Lipi dataset and achieved **94.74%** accuracy [13]. Rahman et al., in 2020, applied a two steps approach to recognize 36 classes and achieved **95.83%** accuracy [30]. Correspondingly, CNN with data augmentation technique is employed to achieved satisfactory performance accuracy from BSL recognition [31,32]. In contrast, the evaluated size of the dataset is still inadequate and they used only one dataset which cannot confirms the effectiveness of the model. To solve the insufficient dataset problem, Rafi et al. [11] built the 38 classes open-source dataset for BSL with an efficient number of images, and the name of the dataset is 38-BdSL which contains 12,160 samples. Then they applied VGG19 architecture to recognize sign language and achieved **89.6%** accuracy. To improve the performance accuracy Abedin et al. proposed a concatenated CNN and achieved **91.5%** accuracy for the 38 BdSL dataset evaluation [33]. However, it is difficult to implement a sign language system using it due to lower accuracy. We did not find a generalized BSL recognition system where the researchers evaluate their model with a different open-source BSL dataset. To overcome the research gap we have developed a novel BenSignNet model to examine the generalizability of the solution for diverse BSL datasets.

2.2. Literary Review on Others Sign Language

Through collaborative research in natural language processing [34,35], computer vision, machine learning [36,37] and pattern matching, sign language recognition has emerged [38,39]. Zimmerman et al. first proposed hand gesture recognition using magnetic flux sensors in 1987 to estimate the hand position and orientation [40]. Yanay et al. worked based on an air-writing recognition using smart bands and achieved 89.2% [41]. Murata et al. proposed the kinetic sensor instead of the smart band and achieved 75.9% average recognition accuracy with dynamic programming (DP) through inter stroke information feature [42]. Besides the kinetic sensor and smart band, Sonoda et al. used the video camera to capture the alphanumeric characters written in the air and

achieved 95% [43]. Although the camera reduces the sensor's complexity, but their work could not determine the starting and ending points for the input and output of the user's hand area in each picture frame. Mukai et al. applied a classification tree and support vector machine (SVM) to recognize the Japanese fingerspelling to solve the problem and achieved 86.0% accuracy [44]. Pariwat et al. applied SVM with different kernels based on local and global features to classify the Thai fingerspelling and achieved satisfiable performance accuracy [45].

Early days, deep learning also used in the other sign language recognition field based on camera images. Ameen et al. used a CNN to recognize fingerspelling images and produced recall of 80% and precision of 82% [46]. For increasing the performance of the sign language recognizer, Nakajai et. used a CNN model-based Histogram of Oriented Gradients (HOG) for recognizing Thai finger spelling images and performed 91.26% precision [47]. Tolentino et al. applied a CNN approach for recognizing American signs through a skin-colour modelling technique and achieve 93.67% accuracy [48]. Hu et al. applied a deep Belief Network (DBN) on the large dataset for ASL recognition and achieved 95.50% accuracy [49]. The reported performance accuracy of this system is good but not enough to implement a real-time system. Aly et al. applied a PCANet for recognizing Arabic sign language(ASL) based on intensity and depth images and achieved 99.5% [50]. In summary, there are many SLR systems with performance accuracy using the corresponding satisfiable size of the datasets. However, we need to develop a BSL recognition system with the efficient size of the BSL dataset.

3. Dataset Description

The actual characters of BSL is 50, among them 30 gestures for alphabet and 10 gestures for digits and commonly used about 4000 double hand symbols [51]. BSL datasets built collaborating with various deaf-community schools and foundations and Table 1 shows the alphabetic representation of BSL characters collected from deaf community schools.

Table 1. Bengali Characters Alphabetic Representation.

অ - A	আ - Ā	ই - I	উ - U	র - R	এ - Ē	ঐ - Oi
ও - Ō	ঔ - Au	ক - K	খ - Kha	গ - Ga	ঘ - Gha	ঙ - Ṅa
চ - Ca	ছ - Cha	জ - Ja	ঝ - Jha	ঞ - Ña	ট - Ṭa	ঠ - Ṭha
ড - Ḍa	ঢ - Ḍha	ত - Ta	থ - Tha	দ - Da	ধ - Dha	ন - Na
প - Pa	ফ - Pha	ব - Ba	ভ - Bha	ম - Ma	য় - Ya	র - Ra
ল - La	স - Sa	হ - Ha	ড় - Ṛa	ঁ - M̐	ং - n	ঃ - Ḥ

However, the experiment is conducted with three individual datasets of the study. These are 38 BdSL, KU-BdSL, Ishara-Lipi described in Sections 3.1–3.3 consequently.

3.1. 38 BdSL Dataset [11]

One of the most benchmark dataset of BSL is 38 BdSL dataset which was collected with the help of the National Federation of the Deaf people. The number of the class included in the dataset is 38 based on the BSL Dictionary published under the Ministry of Social Welfare by the National Center for Special Education. They have collected 320 images for each class and 12,160 images for 38 classes. There were 42 deaf students and 278 non-deaf students who created the dataset. Figure 1 illustrates an example of a sign image for all classes.

Figure 1. Example of BSL Image from 38 BdSL Dataset.

3.2. KU-BdSL [12]

Two variants of the KU-BdSL dataset used in the experiment: Uni-scale sign language dataset (USLD) and the Multi-scale sign language dataset (MSLD). This dataset was collected from 33 participants; 8 were female, and 25 were male. Each variant of the dataset contains 30 classes and 1500 images; all are taken using multiple smartphone cameras, representing single-hand gestures for BdSL. Each image was collected from different subjects and backgrounds with 512 × 512 pixels in size. The intended hand position is placed in the middle of most cases in this dataset. Images are captured from different backgrounds. Figure 2 depicts the sample of images of the KU-BdSL dataset.

Figure 2. Example of KU BdSL Dataset Images.

3.3. Ishara-Lipi Dataset [13]

The Ishara-Lipi dataset contains 1800 images for 36 classes, including 30 consonants and six vowels.

They have collected images from the deaf school community, and all the images have a white background. Each class contains about 50–56 images, with an average of 50 images per class. Each image is resized to 128 × 128 pixels. Figure 3 shows the example of Ishara-Lipi dataset images.

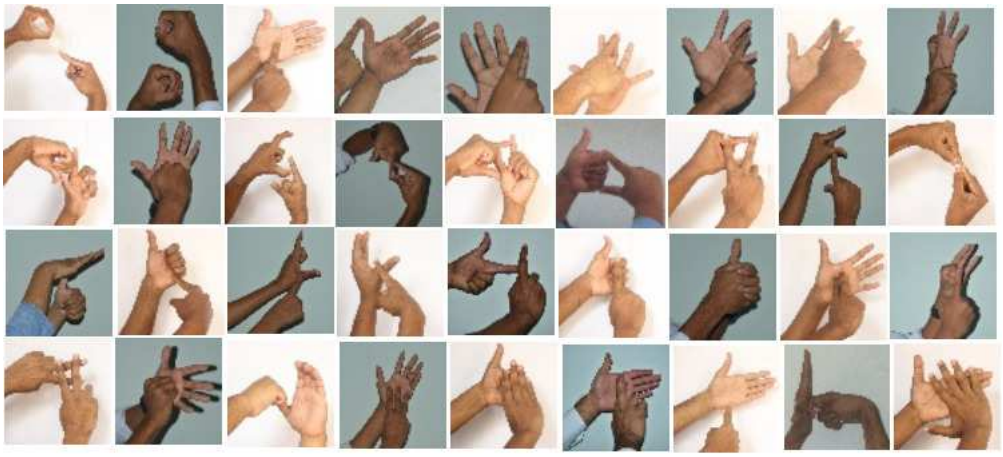

Figure 3. Example of Ishara-Lipi Dataset Images.

4. Proposed System

Figure 4 presents the basic block diagram of the proposed BSL recognition system. We divided this system has four parts, (i) Preprocessing (ii) Segmentation, (iii) Augmentation, (iv) Classification of BSL. The overall process of the proposed methodology is described below:

i Input images are resized to 124 × 124 from the original images, and, therefore, the images are divided into training and test datasets

ii Concatenate segmentation technique applied to remove redundant background.

iii The augmentation technique was applied to the training dataset to increase the size of the dataset without changing the semantic meaning.

iv A novel BenSignNet model is proposed for feature extraction and classification. This model is evaluated with the three datasets mentioned above.

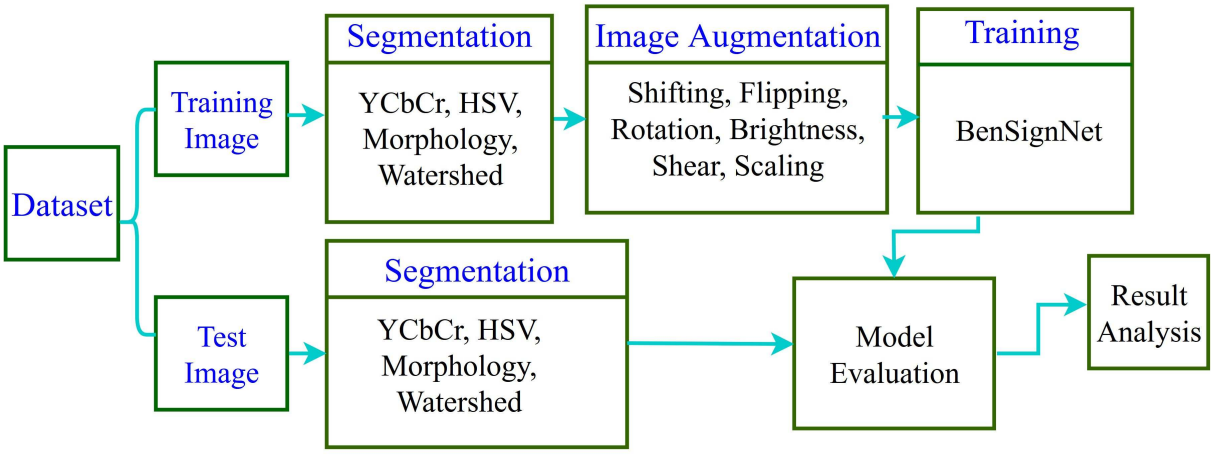

Figure 4. The Overall Architecture of the Proposed BSL Recognition System.

4.1. Segmentation

The study proposes a concatenated segmentation for preprocessing the input image to enhance skin color recognition. This approach was desined by combining YCbCr, HSV, morphology, and the watershed algorithm, which is also used to simplify the computing by eliminating redundant and difficult background concerns. Figure 5 shows the block diagram of the concatenated segmentation method. It contains the following steps:

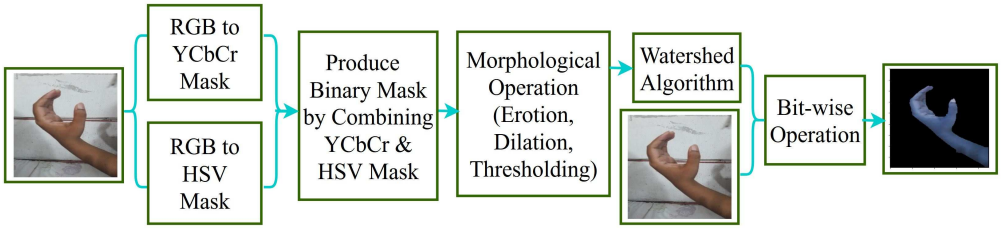

Figure 5. Concatenated Segmentation Techniques.

4.1.1. Binary Mask from YCbCr and HSV

Firstly, images are converted from RGB to YCbCr and HSV [52,53]. YCbCr consists of three components: luminance, blue, and red difference. Also, three-component in the HSV: hue, saturation, and value. The mask frame of the HSV is produced with the upper and lower range of the arguments. Each skin pixel value of the HSV and YCbCr are measured in a standard range for comparing with the pixel quality. Finally, the output of HSV and YCbCr are combined into the binary masks with the Equation (1).

$$BM(m,n) = HSV_{out}(m,n) + YCbCr_{out}(m,n) \quad (1)$$

Here, BM defiend the binary mask, m and n defiend the index of of the image data. The mask is the output of a combination of two parts. Weighted skin produced a white area pixel in the front and a black pixel considered as the background of the mask.

4.1.2. Morphological Operation

Secondly, we apply morphological operation on the output of Section 4.1.1 to make the foreground and background image using erotion and dilation sequentially. Mainly erotion and dilation remove small regions representing a small false-positive skin region. The erotion applied here to produce foreground pixels to foreground pixels may shrink. However, erosion makes foreground images by reducing the noise. The background region is reduced a little because of the dilates operation. Two iterations were used here for both dilation and erotion procedure, and both contained black and white colour [54]. The value of the grayscale image is 0–255, where white represents 255 and black represent zero. To make foreground, we have applied a threshold with 128 values then redefined the pixel value zero for 0–127 and 255 for 128–255 [55]. Then combined both foreground and background, forming markers.

4.1.3. Watershed Algorithm

Thirdly, we applied the watershed algorithm following the process of Sections 4.1.1 and 4.1.2, on the markers, the "seeds" of the future image regions. The watershed algorithm is another region-based algorithm taken from mathematical morphology which marked around the hand gesture of the image [56]. Finally, a bitwise mask is employed be-tween the watershed generated mask image and the input images to produce a segmented image.

4.2. Augmentation Techniques

Every year, many new deep learning architectures are created to achieve state-of-the-art performance accuracy in sign language recognition and image classification. However, a large dataset is still needed to train the BenSignNet models accurately for getting state-of-the-art performance accuracy. In the paper, seven augmentation techniques are used to produce varied image diversity from the input image in order to increase the dataset and make the model image diversity invariant by dealing with the inadequate training of the deep learning model [57]. As augmentation technique, geometric and intensity transformation is applied with different ranges and investigated the impact of each method in model performance [31].

It is observed that new transformation includes extra training time, requiring appropriate strategies. Scaling, translation, shearing and rotation are used here as a geometric transformation which is indicated the mapping medium of the original image pixel position to the generated image position. Additionally, intensity transformation is employed here to modify the pixel value of the image by changing the colour condition and brightness. Table 2 shows the proposed transformation and its possible range for the presented method. The range of each augmentation mentioned in the table is chosen by experimental test and observing image of the dataset. For instance, mentioned range of the rotation, shift, and shear can not be greater than those for the dataset because it may be changed the semantic meaning and partially distort the augmented image.

Table 2. Augmentation Techniques and the Possible Ranges.

Augmentation Technique	Range
Zoom	0.5–1.0
Brightness range	0.2–1.0
Rotation	0–30 degree
Shear	0–10 degree
Width shift range	0.2
Height shift range	0.5
flip	True

4.3. Feature Extraction and Classification Techniques

Deep learning is a part of the broader area of the machine learning approach based on an artificial neural network with four or more layers. Generally, the Deep learning technique has been used to extract features from images and classify those images depending on those extracted features. In the study, a deep learning-based robust CNN architecture namely BenSignNet is designed for BSL recognition.

4.3.1. Basic Concepts of Convolutional Neural Network (CNN)

Usually, the traditional CNN is a feed-forward neural network identical to ordinary neural networks like multilayer perception. CNN architecture includes several convolutional layers, pooling layers, regularizations layers and some functions such as activation and loss functions. The description with the mathematical formula of each layer of uses CNN is described below [10,58,59].

Convolutional Layer

The convolutional layer produces the feature map by performing the convolution between the $n \times n \times d$ dimension input image and $k \times k$ sizes M kernel filters where n defined height, width and d defined the depth of the image. This multiplication is performed pixel by pixel by travelling the filter from left to right of the input images, and zero padding is added around the image pixel to protect the shrinkage of the original input size. The feature map calculation procedure of the convolutional layer is performed according to Equation (2)

$$G_x^{(l)} = \sum_{y=1}^{m_1(l-1)} F_{x,y}^{(l)} \times G_y^{(l-1)} + Bias^{(l)} \quad (2)$$

where $G_x^{(l)}$ defined output feature with x_{th} feature map of layer l, and filter $F(l)$ are matrices connecting with the y_{th} feature map, m_1 defined the kernel, and $G_y^{(l-1)}$ defined the input feature.

Pooling layer

The pooling layer segments the $n \times n$ feature map into the n segment, and in the study, we have employed the two kinds of pooling layer, namely, max-pooling layer, global average pooling layer. Max-pooling layer selects the maximum value for each segment to compress the extracted feature. This used pooling 2 × 2 and stride 2 for making sliding window to skip the width and height with the following Equation (3).

$$\left[\frac{n+2P-f}{2}+1\right] \tag{3}$$

Global average pooling (GAP) [60,61] layer performs dimensionality reduction instead of a fully connected layer, where a tensor with dimensions $n \times n \times d$ is reduced in size to have dimensions $1 \times 1 \times d$. GAP layers reduce each $n \times n$ feature map to a single number by simply taking the average of all (n, n) values. It solves the overfitting problems and increases the generalization property of a model.

Overfitting and Underfitting Control Layers

To reduce the overfitting of the model, we have utilized here dropout and batch normalization layer [62] as a regularizer. The dropout layer identifies the ignoring neurons during the training phase of randomly chosen neuron sets based on the value of the probability. Ignoring neurons do not pass during the forward or backwards pass of the model.

The batch normalization layer calculated the mini-batch from each trial to reduce the training of the model, which is computed by Equation (4)

$$\widetilde{G_x^{(l)}} = \gamma \widehat{G_x^{(l)}} + \beta \tag{4}$$

where $\widetilde{G_x^{(l)}}$ = normalized batch output, $\widehat{G_x^{(l)}} = \frac{G_x^{(l-1)}-\mu}{\sqrt{\sigma^2-\epsilon}}$, then μ, σ^2, ϵ define the average, variance and standard error of the feature map. Correspondingly, γ and β are newly-introduced learnable parameters that scale and shift the normalized values.

Activation and Loss Function

The activation layer is used to normalize the output of the convolution layer. ReLU is computed according to non-linearity function Equation (5)

$$G_x^{(l)} = f\left(G_x^{(l-1)}\right) \tag{5}$$

where l defined the non-linearity layer and normalized feature $G_x^{(l)}$ is generated from the feature map $G_x^{(l-1)}$ of the previous layer $(l-1)$. The main activity of this layer is to set zero for the negative value and return the maximum value according to Equation (6)

$$()G_x^{(l)} = max\left(0, G_x^{(l-1)}\right) \tag{6}$$

The loss function categorical cross-entropy (CCE) [63] produces a high loss value when the actual observation is 1, but the predicting probability is 0.011 and is considered a bad result. Minimized score and perfect cross-entropy value are 0. In our study, we considered an M-pairs training set consisting with: $\{(x_1, c_1), (x_2, c_2), \ldots, (x_M, c_M)\}$, where $x_j \in R^M$ defined the M dimensional jth input vector, $c_j \in R^N$ defined the corresponding targeted

class and $y_j \in R^N$ defined the output. In our study, CCE loss calculated for M-dimensional input x_j to one of N class categories using (7).

$$Loss = -\frac{1}{M}\sum_{j=1}^{M}\sum_{c=1}^{N} y_{j,c}\log(p_{j,c}) \tag{7}$$

where N defined the number of classes, $y_{j,c}$ defined binary indicator function if the class label c is the correct classification for jth training observation. $p_{j,c}$ defined the predicted probability for $j$the training observation is of the class label c. The target $y_{j,c}$ able to be compiled as true and the predicted probability output $p_{j,c}$ for the jth observation belonging to class c. This function mainly computed the average of the sum between the actual probability and the predicted probability for all classes in the study.

Output Layer

The output layer classifies the image into n neurons which represents number of classes of the dataset. The softmax activation function is applied in this layer.

4.3.2. BenSignNet: The Proposed CNN Architecture

Figure 6 shows the proposed BenSignNet model architecture comprises nine convolution layers with a specific filter size, eight ReLU activation layers, two max-pooling layers, a batch normalization layer, a global average pooling layer, and an output layer with a softmax activation function. The output layer is a 38 neuron softmax, which defines the 38 classes. The dropout layer is placed after the convolution layer to reduce the overfitting of the model. The max-pooling layer is placed after the drop out layer to down-sample the value of the convolution feature map. The ReLU activation layer incorporates the each convolution layers. The global average pooling layer is used to replace the fully connected layer.

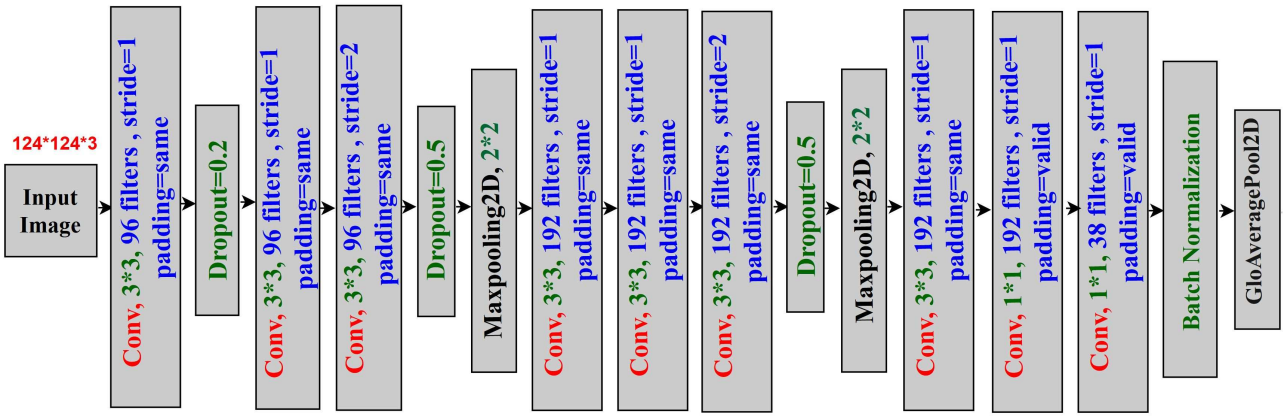

Figure 6. Proposed BenSignNet Architecture.

The input image 124 × 124 × 3 is filtered at the first convolutional layer with 96 kernels of size 3 × 3 × 3. After applying the ReLU activation and dropout layer, the output of the first convolutional layer is fed into the second convolutional layer as input. In second convolution layer filtered the input with the 96 kernels of size 3 × 3 × 96 and passed into the third convolution layer after applying the ReLU function. Third convolutional layers filtered the input with the 96 kernels of size 3 × 3 × 96 based on the 2 stride and fed the ouput at the max-pooling layer.

The fourth convolution layer takes as input the pooled output layer and filters it with 192 kernels of size 3 × 3 × 96, which fed in the fifth convolution layer by filtering it with 192 filters fo size 3 × 3 × 192. After applying the relu activation on the fifth output layer, it fed

into the sixth convolution layer, whose stride is two and filtered with 192 kernels of size 3 × 3 × 192.

The seventh convolutional layer takes as input the pooled output of the six convolutional layers and filters the feature map with 192 kernels in size of 3 × 3 × 192. After applying the ReLU on the output of seventh convolutional layer its fed into the eigth convolution layer, which is filtered with the 192 kernels of size 1 × 1 × 192. The output of the eighth convolutional layer is fed into the ninth convolutional layer after applying the ReLU activation and filtered the feature map with the 38 kernels in size of 1 × 1 × 192. The value of the stride is 2 for the third and sixth convolutional layers and 1 for the rest convolutional layer.

The pool size is 2, and the stride is 2 for both max-pooling layers after the third and sixth convolutional layers.

The dropouts and global average pooling technique are used to reduce the overfitting of the model. Improving the generalization property and solving the layer's overfitting issue are the main advantages of the global average pooling because there is no parameter to optimize in the global average pooling. In addition, global average polling is more robust to spatial translations of the input than the fully connected layer because which sums out the spatial information. Table 3 depicts the details specification of the BenSignNet Model.

Finally, the model is compiled and optimized with the Adam optimizer using the categorical cross-entropy loss function. Our actual output is converted into the one-hot-encoded, making them directly fitted with the categorical cross-entropy loss function.

Table 3. Details Specification of the BenSignNet model.

Layer No	Layer Name	Input Shape	Output Shape	Param
1	Conv2d_1	124 × 124 × 3	124 × 124 × 96	2688
2	Dropout_1	124 × 124 × 96	124 × 124 × 96	0
3	Conv2d_2	124 × 124 × 96	124 × 124 × 96	83,040
4	Conv2d_3	124 × 124 × 96	62 × 62 × 96	83,040
5	Dropout_2	62 × 62 × 96	62 × 62 × 96	0
6	Max Pooling 2d_1	62 × 62 × 96	31 × 31 × 192	0
7	Conv2d_4	31 × 31 × 192	31 × 31 × 192	166,080
9	Conv2d_5	31 × 31 × 192	31 × 31 × 192	331,968
10	Conv2d_6	31 × 31 × 192	16 × 16 × 192	331,968
11	Dropout_3	16 × 16 × 192	16 × 16 × 192	0
12	Max Pooling 2d_2	16 × 16 × 192	8 × 8 × 192	0
13	Conv2d_7	8 × 8 × 192	8 × 8 × 192	331,968
14	Activation (Relu)	8 × 8 × 192	8 × 8 × 192	0
15	Conv2d_8	8 × 8 × 192	8 × 8 × 192	37,056
16	Activation (Relu)	8 × 8 × 192	8 × 8 × 192	0
17	Conv2d_9	8 × 8 × 192	8 × 8 × 38	7334
18	Batch Normalization	8 × 8 × 38	8 × 8 × 38	152
19	Global Average Pooling 2D	8 × 8 × 38	38	0
20	Activation (Softmax)	38	38	0
	Total params: 1,375,294 Trainable params: 1,375,218 Non-trainable params: 76			

5. Result and Discussion

As described in Section 3, the three benchmark BSL datasets have been used to evaluate the proposed method's performance and conduct the experiments. In Section 5.1, we have described the experimental setup, and in Sections 5.3–5.5 described the performance evaluation of 38 BdSL, KU-BdSL and Ishara-Lipi datasets consequently.

5.1. Experimental Setup

For implementing the experiment, The Python programming language is used to implement the proposed experiment on a Google Colab Pro edition environment with a 25 GB GPU called Tesla P100. Cv2, NumPy, Pickle, TensorFlow, Keras, Matplotlib, are used as the python package. In addition, learning rates 0.001, 38 and 36 and 30 classes and adam optimizer used in the CNN architecture.

For experiment, the dataset is divided into training and testing, where **70%** of the dataset is considered as a training dataset and **30%** is regarded as a test dataset. To increase the size of the training dataset, the augmentation technique is applied to the training dataset. Table 4 shows the number of training and testing images for each dataset before and after augmentation. The training dataset of the 38 BdSL, KU-BdSL, and Ishara-Lipi datasets was 8512, 1050, and 1260 images, respectively. After applying the augmentation technique, we have sequentially found 68,096, 15,750 and 18,900 images for the 38 BdSL, KU-BdSL and Ishara-Lipi datasets.

Table 4. Training and Testing Images for Each Dataset.

Dataset	Before Augmentation		After Augmentation	
	Train	Test	Train	Test
38 BdSL [11]	8512	3648	68,096	3648
KU-BdSL [12]	1050	450	15,750	450
Ishara-Lipi [13]	1260	540	18,900	540

5.2. Evaluation Metrics

The performance of the BSL recognition using BenSignNet was measured by accuracy, precision, F1-score and recall, based on the true positive (TP), false positive (FP), true negative (TN) and false negative(FN) defined as below:

The ratio between the total count of correctly predicted classes and total count of sample is called accuracy was computed using Equation (8).

$$Accuracy = \frac{TP + TN}{P + FN + FP + TN} \times 100 \tag{8}$$

The ratio between the total count of correctly predicted positive sign classes and total count of classes is called precision or sensitivity was computed using Equation (9).

$$Precision = \frac{TP}{TP + FP} \times 100 \tag{9}$$

The ratio between the total count of correctly predicted positive classes and total count of positive classes is called precision or sensitivity was computed using Equation (10).

$$Recall = \frac{TP}{TP + FN} \times 100 \tag{10}$$

The harmonic mean between the precision and recall is called the F1 score was calculated by the Equation (11).

$$F1score = \frac{2 \times PrecsionRecall}{Precsion + Recall} \times 100 \tag{11}$$

5.3. Performance Evaluation with 38 BdSL Dataset

In the experiment, the performance of the 38 BdSL datasets is evaluated on the fine-tuned of BenSignNet model with CCE loss function. According to Section 5.1, the test set of the 38 BdSL was **30%**, which is divided equally into validation and test set to compare with the existing method. However, the proposed system with segmented and non-segmented datasets achieved **94%** and **93.20%** accuracy for test dataset. Table 5 shows the performance accuracy of the proposed model for training, validation and test dataset.

Table 5. Training, Validation and Test accuracy of the Proposed System.

Dataset	Segmented	Training (%)	Validation (%)	Testing (%)
38 BdSL alphabets	no	98.00	95.00	93.20
38 BdSL alphabets	yes	99.99	96.00	**94.00**

Figure 7 shows the class-wise alphabet recognition bar chart for our proposed system: precision, recall, f1-score and performance accuracy of the 38 classes of the 38 BdSL alphabets dataset. Our model produced high-performance accuracy in all the classes by observing the class-wise figure, excluding two classes.

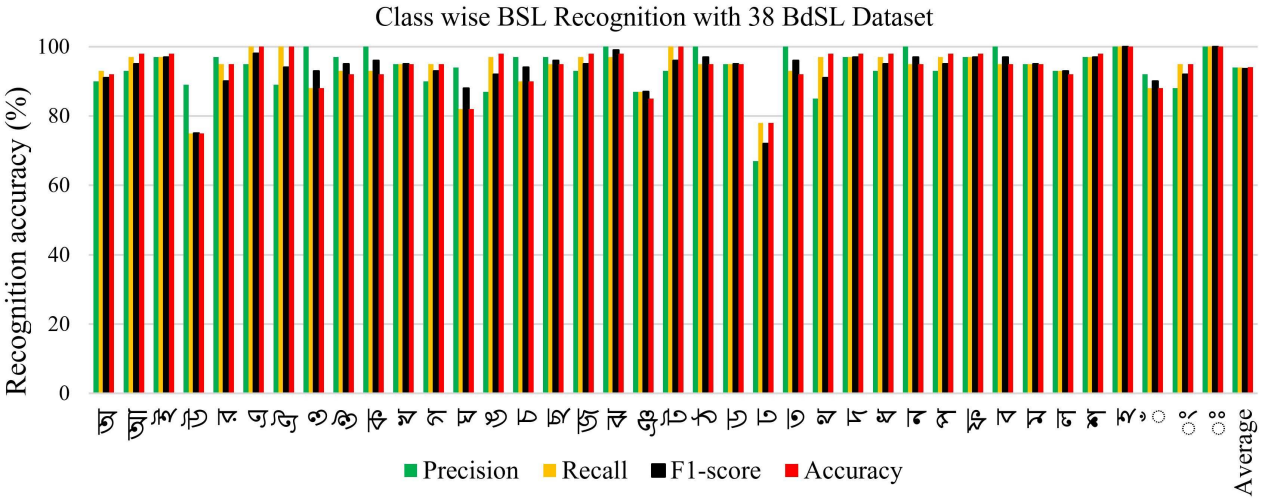

Figure 7. Class wise Precision, Recall, F1-score and Accuracy of the Proposed method .

Confusion matrix of detection of the BdSL alphabet signs using the proposed system shown in Figure 8. Here, probabilities along the main diagonal or correct detection rate are defined as predicted and output classes. Alphabets sign in a predicted class represents each row of the confusion matrix, while each column represents one in a correct class. The proposed model produces good accuracy for all classes and correctly classified almost more than **90%** except four classes. Among them are some misclassification, like **35%** misclassification in (22,3) and **15%** misclassification in (13,18). This happened because some signs are almost similar, so the classifier may face difficulties differentiating them.

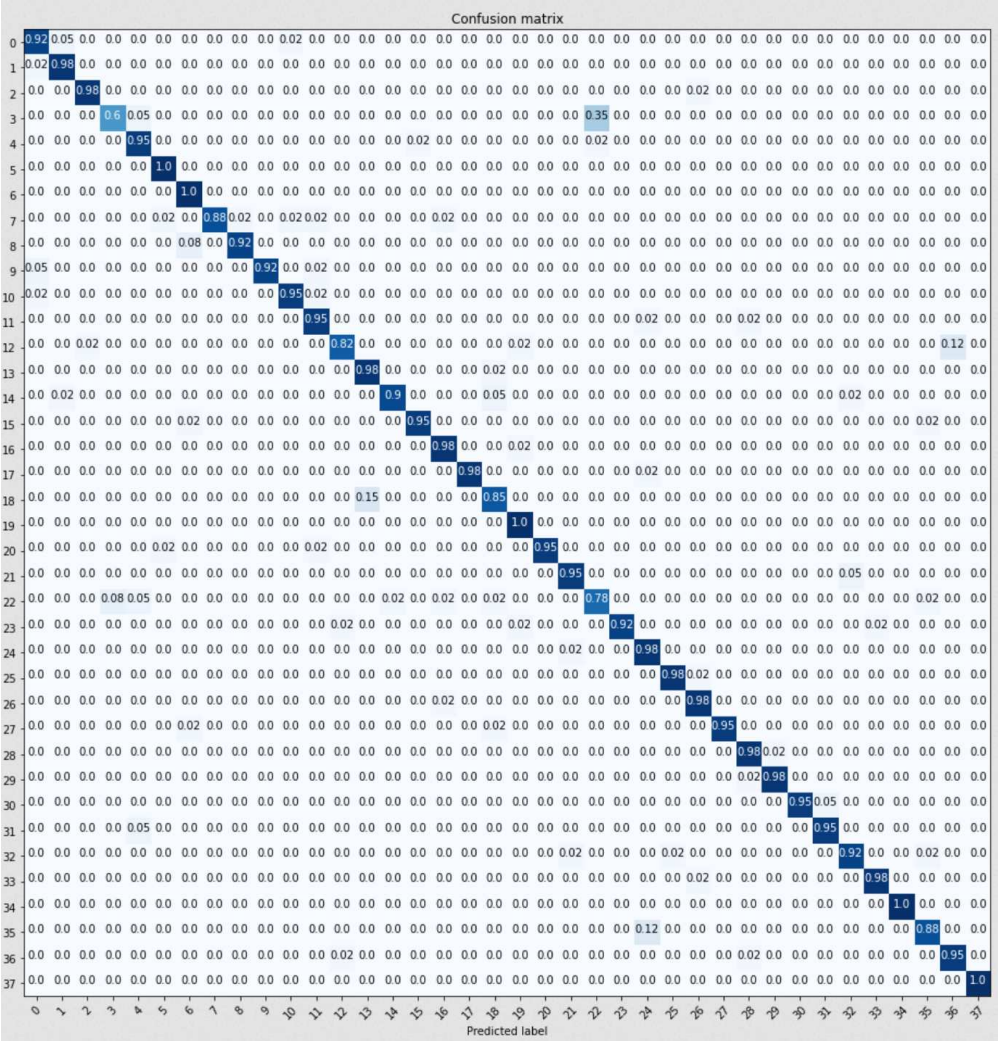

Figure 8. Confusion Matrix of the Proposed Model with 38 BdSL Dataset.

Table 6 shows the comparison accuracy of the proposed method with the existing model using 38 BdSL datasets. The proposed system produced **94.00%** accuracy, which is better than the state-of-the-art method. The authors in [11] used VGG19 based CNN to recognize the BSL alphabets. They have trained their model using SGD optimizer with the learning rate 1 $\times 10^{-3}$ and achieved **89.60%** accuracy.

Table 6. State of the Art Comparison for 38 BdSL Dataset.

Dataset	Model Name	Segmented	Image Pixel	Training (%)	Validation (%)	Testing (%)
38 BdSL	[11]	No	224 × 224	97.68	91.52	89.60
38 BdSL	[33]	no	60 × 60	98.67	95.28	91.52
38 BdSL	Proposed model (BenSignNet)	yes	124 × 124	99.99	96.00	**94.00**

The authors [33], proposed a concatenated BdSL network to recognize the 38 BdSL sign language alphabets and achieved **91.52%** testing accuracy. Firstly, they resized the image in 64 × 64; they selected a CNN architecture to extract the visual feature. They combined the CNN based feature with the hand pose estimation base feature. Their CNN

architecture included ten convolutions, 10 ReLu activation, four max-pooling, and a single input-output layer. In the hand pose estimation, they have calculated the hand key points from the image. Then the CNN and hand pose feature vector passed through two fully connected layers and finally used the softmax activation function.

Figure 9 shows the class-wise comparison accuracy of the proposed model with the existing model. By observing the figure, we can decide that our proposed model produced higher performance accuracy for all the classes than the existing model class wise performance accuracy. Among 38 classes, 35 classes produced more than **90%** accuracy; one class has **80%** accuracy, and only two class made below **80%** accuracy.

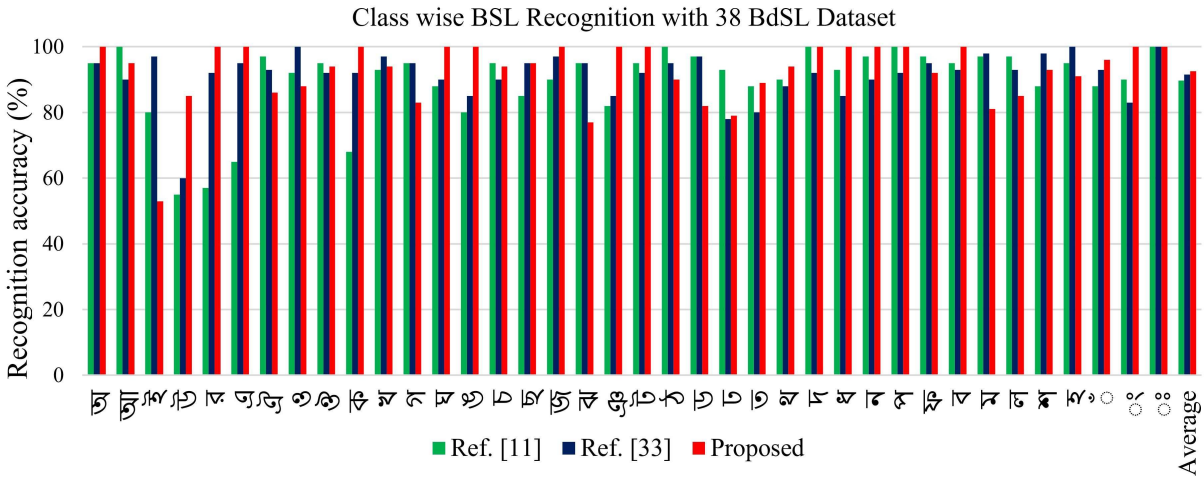

Figure 9. Class wise State of the Art Comparison for 38 BdSL datasets.

5.4. Performance Evaluation with KU-BdSL Dataset

We evaluated the proposed model with two variants of KU-BdSL datasets. The proposed system achieved **98.66%** accuracy for validation and **98.20%** accuracy for testing for the non-segmented dataset. The same accuracy was achieved for validation and testing in the segmentation case, which is **99.60%**. Table 7 shows the performance accuracy of the proposed model for training, validation and test dataset.

Table 7. Performance Accuracy for KU-BdSL USLD Variant Dataset.

Dataset	Segmented	Training (%)	Validation (%)	Testing (%)
KU-BdSL USLD Variant	No	99.10	98.66	98.20
KU-BdSL USLD Variant	Yes	99.90	99.60	**99.60**

Table 8 shows the performance accuracy of the proposed model for training, validation and test dataset. The table shows our system with non-segmented images achieved **98.66%** accuracy for validation and **98.20%** accuracy for the test dataset. It also shows the performance accuracy of the proposed method using a segmented dataset that is **99.99%** accuracy for validation and **99.60%** accuracy for the testing dataset.

Table 8. Performance Accuracy for KU-BdSL MSLD Variant Dataset.

Dataset	Segmented	Training (%)	Validation (%)	Testing (%)
KU-BdSL MSLD Variant	No	99.10	98.66	98.20
KU-BdSL MSLD Variant	Yes	100	99.99	**99.60**

Figure 10 shows the class-wise sign word recognition bar chart for our proposed model with USLD and MSLD variants of the KU-BdSL dataset. By observing the class-wise

performance, we can say our model produced high-performance accuracy for all the classes, excluding three classes for both variants of the KU-BdSL datasets. Our model produces good accuracy for all classes and correctly classified almost more than **99%** except for three classes. The 27 classes of MSLD variant produced more than **99%** accuracy for two classes produced **95%** accuracy, and one class produced **93%** accuracy. For the USLD variant, 27 classes have more than **99%** accuracy, but it produced **97%** accuracy for 'Ja' class and **95%** accuracy for "Ka' and 'Sha' classes.

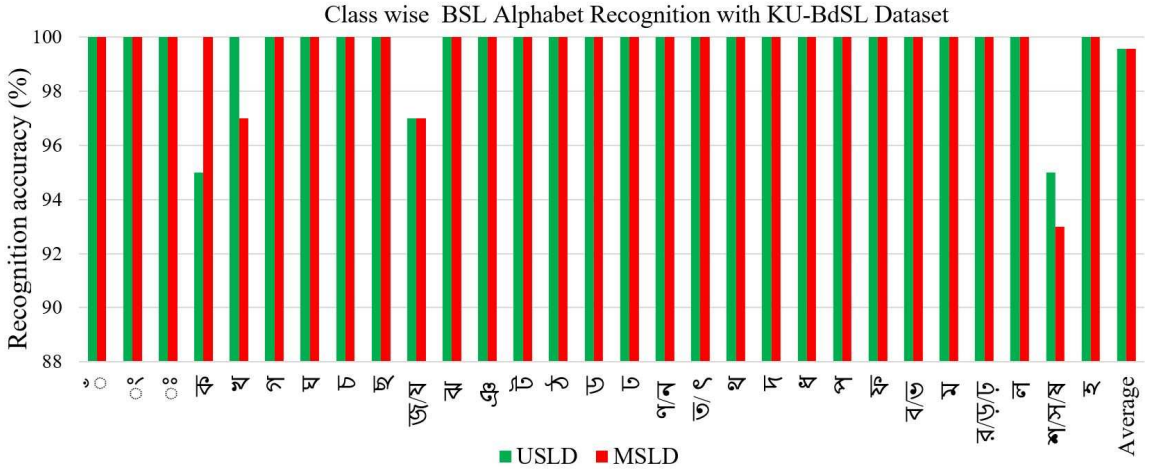

Figure 10. Class wise Performance Accuracy of the Proposed Model with KU-BdSL Dataset.

Table 9 shows the comparison accuracy of the proposed method with the existing model using KU-BdSL datasets. We compared with some relevant work based on the BSL dataset, which shows the proposed model performance is higher than the state of the art model performance. However, due to the new dataset, there is no published paper on the KU-BdSL dataset.

Table 9. State of the Art Comparison for KU-BdSL Dataset.

Reference	Gesture	Sample	Segmen Tation	Pixel	Model	Vectorize	Accuracy (%)
[64]	38	7600	Yes	128 × 128	CNN	FC [a]	90.63
[65]	10	100	No	N/A	R-CNN	FC	98.20
Proposed model	31	3000	yes	124 × 124	Ben SignNet	GAP [b]	**99.60**

[a] FC- Fully connected layer; [b] GAP-Global average pooling layer.

5.5. Performance Evaluation with Ishara-Lipi Dataset

Table 10 shows the performance accuracy of the proposed model with the Ishara-Lipi dataset. The proposed system achieved **99.10%** accuracy for non-segmented images and **99.60%** for the segmented image.

Table 10. Performance Accuracy the Proposed Model with Ishara-Lipi Dataset.

Dataset	Model Name	Segmented	Test Set
Ishara-Lipi	CNN	No	99.10
Ishara-Lipi	CNN	Yes	**99.60**

Table 11 shows the comparison performance accuracy of the proposed model with the state-of-the-art method using the Ishara-Lipi dataset. The result shows that the proposed

model achieved **99.60%** accuracy,which is better than the state-of-the-art model. In [13], the authors used four convolution layers, two max-pooling layers, one dense layer and one dropout layer. Also, an ADM optimizer was used with a learning rate of 0.001 and achieved **94.74%** accuracy.

Table 11. State of the Art Comparison for Ishara-Lipi Dataset.

Dataset	Model Name	Segmented	Image Pixel	Accuracy (%)
Ishara-Lipi	[13]	No	128 × 128	94.74
Ishara-Lipi	[30]	No	N/A	95.83
Ishara-Lipi	[31]	No	N/A	99.20
Ishara-Lipi	Proposed model (BenSignNet)	yes	124 × 124	99.60

The authors in [30], they proposed two-step classifiers for classifying the BSL dataset. As the first step they used 2 phase classifiers: Normalised NOBV and WGV. Initially, they tried to apply NOBV to classify the hand sign, but if the classification score does not satisfy the threshold value, they applied a WGV classifier. They also used the CNN-based Bengali Language Modeling (BLMA) algorithm, achieving **95.83%** accuracy.

In [31], authors proposed a CNN for recognizing BSL 36 alphabets based on the Ishara-Lipi dataset, and they achieved **99.22%** accuracy. They firstly augmented the dataset then produced 1000 images for each class and 36000 images for 36 classes. Then they applied CNN with six convolutional layers, three pooling layers, fully connected layers. They employed translation scaling variance in the pooling layer to decrease the feature volume. Figure 11 shows the class-wise classification accuracy comparison between the proposed and state-of-the-art methods. Which shows, the proposed system produced more than **96%** accuracy for all classes excluded two classes

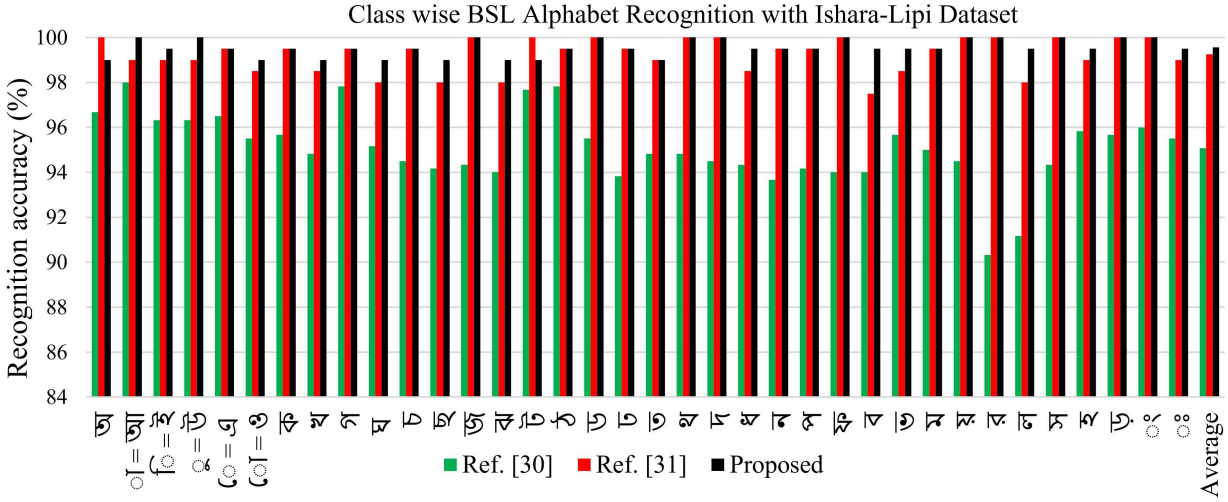

Figure 11. Class wise State of the Art Comparison for Ishara Lipi Dataset.

5.6. Discussion

This paper proposes a generalized system to evaluate the three benchmark datasets on BSL. For this, the input images were segmented for feature extraction and then the images were augmented to increase the dataset for evaluation of the proposed model. However, our proposed model achieved **94%** accuracy with 38 BdSL datasets, as shown in Table 5. Some other researchers evaluated their model using the same dataset. Table 6 shows the comparative performance of the proposed model with the existing model, where one paper

reported **89.60%** accuracy using VGG19 [11] and another article reported **91.50%** accuracy using Concatenated CNN [33]. Figure 9 demonstrates our proposed method's class wise performance comparison with the existing method. Among 38 classes, in 16 classes achieved more than **99%** accuracy, in 19 classes achieved more than **92%** accuracy, in one class achieved **80%** accuracy, and only two made below **80%** accuracy. Class 'I' achieved **57%** accuracy, and class 'Jha' achieved **78%** accuracy. In the same way, our proposed model achieved **99.55%** accuracy with KU-BdSL in both variant datasets, as shown in Table 7 and Table 8. As KU-BdSL is a new dataset, we did not find any published paper based on the KU-BdSL to make a comparison but in Table 9, we show a comparison with other dataset.

Figure 10 demonstrate the class-wise performance accuracy for two variant of the KU-BdSL dataset. Among the 30 classes, 27 classes produced **100%** accuracy for the both variants dataset. Table 11 shows the accuracy comparison of the proposed method with state-of-the-art methods using the Ishara-Lipi Dataset. Our approach with the Ishara-Lipi dataset has achieved **99.60%** accuracy whereas **94.74%** achieved by method [13], **95.83%** accuracy achieved by BLMA [30] and **99.20%** achieved by method [31]. Figure 11 shows the class-wise comparison accuracy between the proposed and the state-of-the-art methods for Ishara-Lipi dataset. Where demonstrated, 12 classes achieved **100%** accuracy and the rest of the 24 classes acquired around **99%** accuracy, which is better than all the state-of-the-art class-wise accuracy. Figure 12 shows the comparison of the proposed method with the state-of-the-art method. The results show that the proposed method is achieved generalized property in the BSL recognition field.

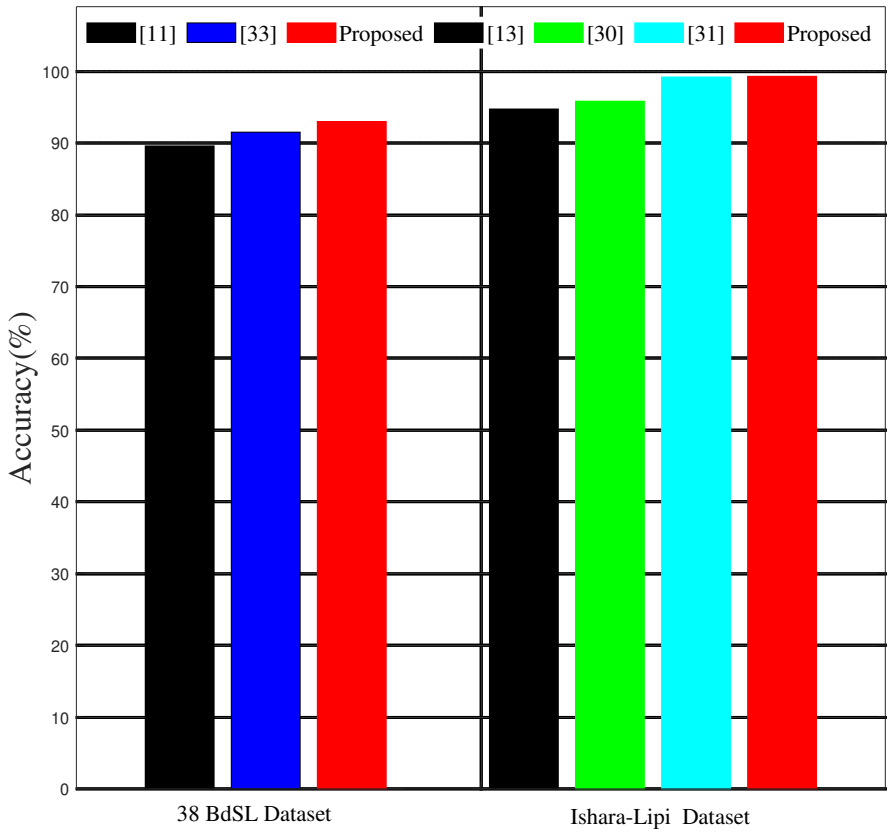

Figure 12. State-of-the-Art Comparison of the Proposed Method (BenSignNet).

6. Conclusions

This paper proposes a general architecture for the BSL alphabet recognition system. The proposed system was completed with three steps: image segmentation, dataset augmentation, and classification techniques. We applied a combined segmentation technique including YCbCr, HSV, and the Watershed algorithm to segment hand gestures from the input images. Then effective augmentation techniques was applied to enlarge the datasets that do not require new images to train the model accurately. A robust BenSignNet model is used for feature extraction and classification. However, three BSL benchmark datasets, such as 38 BdSL, KU-BdSL, Ishara-Lipi, was used to evaluate the effectiveness of the proposed model. As a result, we achieved **94%** recognition accuracy for 38 BdSL, **99.60%** for KU-BdSL, and **99.60%** for the Ishara-Lipi dataset. Finally, we can say that recognition accuracy shows that the proposed generalized system performs better than state-of-the-art methods. Hopefully, this method would become a benchmark for all humanitarian projects serving the deaf and mute community using BSL.

In the future, we look forward to enlarging the dataset by collecting from 1000 signers of different ages people for the BSL and building a larger dataset with a longer length of Bengali sentences. We could invite some professionals working with the deaf and mute community to validate the dataset images. Then we will improve the system and validate with the highly standardized data. After training with the huge amount of standardized datasets, the system could fulfil the requirement to establish communication among deaf and hard of hearing communities.

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

Funding: This work was supported by Competition Research of The University of Aizu, Japan.

Institutional Review Board Statement:

Informed Consent Statement:

Data Availability Statement:

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

References

1. Cheok, M.J.; Omar, Z.; Jaward, M.H. A review of hand gesture and sign language recognition techniques. *Int. J. Mach. Learn. Cybern.* **2019**, *10*, 131–153.
2. Murray, J.; Snoddon, K.; Meulder, M.; Underwood, K. Intersectional inclusion for deaf learners: moving beyond General Comment no. 4 on Article 24 of the United Nations Convention on the Rights of Persons with Disabilities. *Int. J. Incl. Educ.* **2018**, *24*, 1–15. doi:10.1080/13603116.2018.1482013.
3. Tarafder, K.; Akhtar, N.; Zaman, M.; Rasel, M.; Bhuiyan, M.R.; Datta, P. Disabling hearing impairment in the Bangladeshi population. *J. Laryngol. Otol.* **2015**, *129*, 1–10. doi:10.1017/S002221511400348X.
4. Zhang, Z.; Li, Z.; Liu, H.; Cao, T.; Liu, S. Data-driven Online Learning Engagement Detection via Facial Expression and Mouse Behavior Recognition Technology. *J. Educ. Comput. Res.* **2020**, *58*, 63–86. doi:10.1177/0735633119825575.
5. Liu, T.; Liu, H.; Li, Y.F.; Chen, Z.; Zhang, Z.; Liu, S. Flexible FTIR Spectral Imaging Enhancement for Industrial Robot Infrared Vision Sensing. *IEEE Trans. Ind. Inform.* **2020**, *16*, 544–554. doi:10.1109/TII.2019.2934728.
6. Rajan, R.G.; Leo, M.J. American sign language alphabets recognition using hand crafted and deep learning features. In Proceedings of the 2020 International Conference on Inventive Computation Technologies (ICICT), Coimbatore, Tamilnadu, 26–28 February 2020; pp. 430–434.
7. Kudrinko, K.; Flavin, E.; Zhu, X.; Li, Q. Wearable sensor-based sign language recognition: A comprehensive review. *IEEE Rev. Biomed. Eng.* **2020**, *14*, 82–97.

8. Sharma, S.; Singh, S. Vision-based sign language recognition system: A Comprehensive Review. In Proceedings of the 2020 International Conference on Inventive Computation Technologies (ICICT), Coimbatore, Tamilnadu, 26–28 February 2020; pp. 140–144.
9. Podder, K.K.; Chowdhury, M.E.H.; Tahir, A.M.; Mahbub, Z.B.; Khandakar, A.; Hossain, M.S.; Kadir, M.A. Bangla Sign Language (BdSL) Alphabets and Numerals Classification Using a Deep Learning Model. *Sensors* **2022**, *22*, 574.
10. Awan, M.J.; Rahim, M.S.M.; Salim, N.; Rehman, A.; Nobanee, H.; Shabir, H. Improved Deep Convolutional Neural Network to Classify Osteoarthritis from Anterior Cruciate Ligament Tear Using Magnetic Resonance Imaging. *J. Pers. Med.* **2021**, *11*, 1163.
11. Rafi, A.M.; Nawal, N.; Bayev, N.S.; Nima, L.; Shahnaz, C.; Fattah, S.A. Image-based bengali sign language alphabet recognition for deaf and dumb community. In Proceedings of the 2019 IEEE global humanitarian technology conference (GHTC), Seattle, WA, USA, 17–20 October 2019; pp. 1–7.
12. Rafi, A.M.; Nawal, N.; Bayev, N.S.N.; Nima, L.; Shahnaz, C.; Fattah, S.A. KU-BdSL: Khulna University Bengali Sign Language dataset. *Mendeley Data* **2021**, *1*. doi:10.17632/scpvm2nbkm.1.
13. Islam, M.S.; Mousumi, S.S.S.; Jessan, N.A.; Rabby, A.S.A.; Hossain, S.A. Ishara-lipi: The first complete multipurposeopen access dataset of isolated characters for bangla sign language. In Proceedings of the 2018 International Conference on Bangla Speech and Language Processing (ICBSLP), Sylhet, Bangladesh, 21–22 September 2018; pp. 1–4.
14. Hoque, M.T.; Rifat-Ut-Tauwab, M.; Kabir, M.F.; Sarker, F.; Huda, M.N.; Abdullah-Al-Mamun, K. Automated Bangla sign language translation system: Prospects, limitations and applications. In Proceedings of the 2016 5th International Conference on Informatics, Electronics and Vision (ICIEV), Dhaka, Bangladesh, 13–14 May 2016; pp. 856–862.
15. Islalm, M.S.; Rahman, M.M.; Rahman, M.H.; Arifuzzaman, M.; Sassi, R.; Aktaruzzaman, M. Recognition Bangla Sign Language using Convolutional Neural Network. In Proceedings of the 2019 International Conference on Innovation and Intelligence for Informatics, Computing, and Technologies (3ICT), Sakhier, Bahrain, 22–23 September 2019; pp. 1–6. doi:10.1109/3ICT.2019.8910301.
16. Liu, H.; Liu, T.; Zhang, Z.; Sangaiah, A.K.; Yang, B.; Li, Y.F. ARHPE: Asymmetric Relation-aware Representation Learning for Head Pose Estimation in Industrial Human-machine Interaction. *IEEE Trans. Ind. Inform.* **2022**. doi:10.1109/TII.2022.3143605.
17. Liu, H.; Nie, H.; Zhang, Z.; Li, Y.F. Anisotropic angle distribution learning for head pose estimation and attention understanding in human-computer interaction. *Neurocomputing* **2021**, *433*, 310–322. doi:10.1016/j.neucom.2020.09.068.
18. Liu, H.; Fang, S.; Zhang, Z.; Li, D.; Lin, K.; Wang, J. MFDNet: Collaborative Poses Perception and Matrix Fisher Distribution for Head Pose Estimation. *IEEE Trans. Multimed.* **2021**. doi:10.1109/TMM.2021.3081873.
19. Li, Z.; Liu, H.; Zhang, Z.; Liu, T.; Xiong, N.N. Learning Knowledge Graph Embedding With Heterogeneous Relation Attention Networks. *IEEE Trans. Neural Netw. Learn. Syst.* **2021**, 1–13. https://doi.org/10.1109/TNNLS.2021.3055147.
20. Liu, H.; Zheng, C.; Li, D.; Shen, X.; Lin, K.; Wang, J.; Zhang, Z.; Zhang, Z.; Xiong, N.N. EDMF: Efficient Deep Matrix Factorization with Review Feature Learning for Industrial Recommender System. *IEEE Trans. Ind. Inform.* **2021**. doi:10.1109/TII.2021.3128240.
21. Liu, H.; Zheng, C.; Li, D.; Zhang, Z.; Lin, K.; Shen, X.; Xiong, N.N.; Wang, J. Multi-perspective social recommendation method with graph representation learning. *Neurocomputing* **2022**, *468*, 469–481. doi:10.1016/j.neucom.2021.10.050.
22. Kaushik Deb, D.; Khan, M.I.; Mony, H.P.; Chowdhury, S. Two-handed sign language recognition for bangla character using normalized cross correlation. *Glob. J. Comput. Sci. Technol.* **2012**, *12*, 1–7.
23. Karmokar, B.C.; Alam, K.M.R.; Siddiquee, M.K. Bangladeshi sign language recognition employing neural network ensemble. *Int. J. Comput. Appl.* **2012**, *58*, 43–46.
24. Rahaman, M.A.; Jasim, M.; Ali, M.H.; Hasanuzzaman, M. Real-time computer vision-based Bengali sign language recognition. In Proceedings of the 2014 17th International Conference on Computer and Information Technology (ICCIT), Dhaka, Bangladesh, 22–23 December 2014; pp. 192–197.
25. Rahaman, M.A.; Jasim, M.; Ali, M.H.; Hasanuzzaman, M. Computer vision based bengali sign words recognition using contour analysis. In Proceedings of the 2015 18th International Conference on Computer and Information Technology (ICCIT), Dhaka, Bangladesh, 21–23 December 2015; pp. 335–340.
26. Uddin, M.A.; Chowdhury, S.A. Hand sign language recognition for bangla alphabet using support vector machine. In Proceedings of the 2016 International Conference on Innovations in Science, Engineering and Technology (ICISET), Dhaka, Bangladesh, 28–29 October 2016; pp. 1–4.
27. Yasir, F.; Prasad, P.W.C.; Alsadoon, A.; Elchouemi, A.; Sreedharan, S. Bangla Sign Language recognition using convolutional neural network. In Proceedings of the 2017 International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT), Kannur, India, 6–7 July 2017; pp. 49–53.
28. Hoque, O.B.; Jubair, M.I.; Islam, M.S.; Akash, A.F.; Paulson, A.S. Real time bangladeshi sign language detection using faster r-cnn. In Proceedings of the 2018 international conference on innovation in engineering and technology (ICIET), Dhaka, Bangladesh, 27–28 December 2018; pp. 1–6.
29. Islam, M.S.; Sultana Sharmin, S.; Jessan, N.; Rabby, A.S.A.; Abujar, S.; Hossain, S. Ishara-Bochon: The First Multipurpose Open Access Dataset for Bangla Sign Language Isolated Digits. In *Recent Trends in Image Processing and Pattern Recognition, Proceedings of the International Conference on Recent Trends in Image Processing and Pattern Recognition, Solapur, India, 21–22 December 2019*; Springer: Singapore, 2019.
30. Rahaman, M.A.; Jasim, M.; Ali, M.; Hasanuzzaman, M. Bangla language modeling algorithm for automatic recognition of hand-sign-spelled Bangla sign language. *Front. Comput. Sci.* **2020**, *14*, 1–20.

31. Hasan, M.M.; Srizon, A.Y.; Hasan, M.A.M. Classification of Bengali sign language characters by applying a novel deep convolutional neural network. In Proceedings of the 2020 IEEE Region 10 Symposium (TENSYMP), Dhaka, Bangladesh, 5–7 June 2020; pp. 1303–1306.
32. Urmee, P.P.; Al Mashud, M.A.; Akter, J.; Jameel, A.S.M.M.; Islam, S. Real-time bangla sign language detection using xception model with augmented dataset. In Proceedings of the 2019 IEEE International WIE Conference on Electrical and Computer Engineering (WIECON-ECE), Bangalore, India, 15–16 November 2019; pp. 1–5.
33. Abedin, T.; Prottoy, K.S.; Moshruba, A.; Hakim, S.B. Bangla sign language recognition using concatenated BdSL network. *arXiv* **2021**, arXiv:2107.11818.
34. Zhang, Z.; Li, Z.; Liu, H.; Xiong, N.N. Multi-scale Dynamic Convolutional Network for Knowledge Graph Embedding. *IEEE Trans. Knowl. Data Eng.* **2020**, *34*, 2335–2347. doi:10.1109/TKDE.2020.3005952.
35. Farooq, U.; Mohd Rahim, M.S.; Khan, N.S.; Rasheed, S.; Abid, A. A Crowdsourcing-Based Framework for the Development and Validation of Machine Readable Parallel Corpus for Sign Languages. *IEEE Access* **2021**, *9*, 91788–91806. doi:10.1109/ACCESS.2021.3091433.
36. Li, D.; Liu, H.; Zhang, Z.; Lin, K.; Fang, S.; Li, Z.; Xiong, N.N. CARM: Confidence-aware recommender model via review representation learning and historical rating behavior in the online platforms. *Neurocomputing* **2021**, *455*, 283–296. doi:10.1016/j.neucom.2021.03.122.
37. Farooq, U.; Shafry, M.; Rahim, M.; Khan, N.; Hussain, A.; Abid, A. Advances in machine translation for sign language: Approaches, limitations, and challenges. *Neural Comput. Appl.* **2021**, *33*, 14357–14399. doi:10.1007/s00521-021-06079-3.
38. Cheok, M.J.; Omar, Z.; Jaward, M.H. A review of hand gesture and sign language recognition techniques. *Int. J. Mach. Learn. Cybern.* **2019**, *10*, 131–153.
39. Wadhawan, A.; Kumar, P. Sign language recognition systems: A decade systematic literature review. *Arch. Comput. Methods Eng.* **2021**, *28*, 785–813.
40. Zimmerman, T.G.; Lanier, J.; Blanchard, C.; Bryson, S.; Harvill, Y. A hand gesture interface device. In Proceedings of the CHI'86 Conference Proceedings, Boston, MA, USA, 13–17 April 1986.
41. Yanay, T.; Shmueli, E. Air-writing recognition using smart-bands. *Pervasive Mob. Comput.* **2020**, *66*, 101183.
42. Murata, T.; Shin, J. Hand gesture and character recognition based on kinect sensor. *Int. J. Distrib. Sens. Netw.* **2014**, *10*, 278460.
43. Sonoda, T.; Muraoka, Y. A letter input system based on handwriting gestures. *Electron. Commun. Jpn. (Part III Fundam. Electron. Sci.)* **2006**, *89*, 53–64.
44. Mukai, N.; Harada, N.; Chang, Y. Japanese fingerspelling recognition based on classification tree and machine learning. In Proceedings of the 2017 Nicograph International (NicoInt), Kyoto, Japan, 2–3 June 2017; pp. 19–24.
45. Pariwat, T.; Seresangtakul, P. Thai finger-spelling sign language recognition using global and local features with SVM. In Proceedings of the 2017 9th International Conference on Knowledge and Smart Technology (KST), Chonburi, Thailand, 1–4 February 2017; pp. 116–120.
46. Ameen, S.; Vadera, S. A convolutional neural network to classify American Sign Language fingerspelling from depth and colour images. *Expert Syst.* **2017**, *34*, e12197.
47. Nakjai, P.; Katanyukul, T. Hand sign recognition for thai finger spelling: An application of convolution neural network. *J. Signal Process. Syst.* **2019**, *91*, 131–146.
48. Tolentino, L.K.S.; Juan, R.O.S.; Thio-ac, A.C.; Pamahoy, M.A.B.; Forteza, J.R.R.; Garcia, X.J.O. Static sign language recognition using deep learning. *Int. J. Mach. Learn. Comput.* **2019**, *9*, 821–827.
49. Hu, Y.; Zhao, H.F.; Wang, Z.G. Sign language fingerspelling recognition using depth information and deep belief networks. *Int. J. Pattern Recognit. Artif. Intell.* **2018**, *32*, 1850018.
50. Aly, S.; Osman, B.; Aly, W.; Saber, M. Arabic sign language fingerspelling recognition from depth and intensity images. In Proceedings of the 2016 12th International Computer Engineering Conference (ICENCO), Cairo, Egypt, 28–29 December 2016; pp. 99–104.
51. Youme, S.K.; Chowdhury, T.A.; Ahamed, H.; Abid, M.S.; Chowdhury, L.; Mohammed, N. Generalization of Bangla Sign Language Recognition Using Angular Loss Functions. *IEEE Access* **2021**, *9*, 165351–165365.
52. Kolkur, S.; Kalbande, D.; Shimpi, P.; Bapat, C.; Jatakia, J. Human Skin Detection Using RGB, HSV and YCbCr Color Models. In Proceedings of the Proceedings of the International Conference on Communication and Signal Processing 2016 (ICCASP 2016), Lonere, India, 26–27 December 2016. doi:10.2991/iccasp-16.2017.51.
53. Saxen, F.; Al-Hamadi, A. Color-based skin segmentation: An evaluation of the state of the art. In Proceedings of the 2014 IEEE International Conference on Image Processing (ICIP), Paris, France, 27–30 October 2014; pp. 4467–4471.
54. Tolentino, L.K.S.; Juan, R.O.S.; Thio-ac, A.C.; Pamahoy, M.A.B.; Forteza, J.R.R.; Garcia, X.J.O. Static sign language recognition using deep learning. *Int. J. Mach. Learn. Comput.* **2019**, *9*, 821–827.
55. Rahim, M.A.; Islam, M.R.; Shin, J. Non-touch sign word recognition based on dynamic hand gesture using hybrid segmentation and CNN feature fusion. *Appl. Sci.* **2019**, *9*, 3790.
56. Kornilov, A.S.; Safonov, I.V. An Overview of Watershed Algorithm Implementations in Open Source Libraries. *J. Imaging* **2018**, *4*, 123. doi:10.3390/jimaging4100123.

57. Carneiro, A.C.; Silva, L.B.; Salvadeo, D.P. Efficient sign language recognition system and dataset creation method based on deep learning and image processing. In Proceedings of the Thirteenth International Conference on Digital Image Processing (ICDIP 2021), Singapore, 20–23 May 2021; Volume 11878, p. 1187803.
58. Fregoso, J.; Gonzalez, C.I.; Martinez, G.E. Optimization of Convolutional Neural Networks Architectures Using PSO for Sign Language Recognition. *Axioms* **2021**, *10*, 139.
59. Jagtap, S.; Bhatt, C.; Thik, J.; Rahimifard, S. Monitoring Potato Waste in Food Manufacturing Using Image Processing and Internet of Things Approach. *Sustainability* **2019**, *11*, 3173.
60. Shustanov, A.; Yakimov, P. Modification of single-purpose CNN for creating multi-purpose CNN. *J. Phys. Conf. Ser.* **2019**, *1368*, 052036. doi:10.1088/1742-6596/1368/5/052036.
61. Rusiecki, A. Trimmed categorical cross-entropy for deep learning with label noise. *Electron. Lett.* **2019**, *55*, 319–320. doi:10.1049/el.2018.7980.
62. Sledevic, T. Adaptation of Convolution and Batch Normalization Layer for CNN Implementation on FPGA. In Proceedings of the 2019 Open Conference of Electrical, Electronic and Information Sciences (eStream), Vilnius, Lithuania, 25 April 2019; pp. 1–4. doi:10.1109/eStream.2019.8732160.
63. Lin, M.; Chen, Q.; Yan, S. Network in network. *arXiv* **2013**, arXiv:1312.4400.
64. Shanta, S.S.; Anwar, S.T.; Kabir, M.R. Bangla Sign Language Detection Using SIFT and CNN. In Proceedings of the 2018 9th International Conference on Computing, Communication and Networking Technologies (ICCCNT), Bengaluru, India, 10–12 July 2018; pp. 1–6. doi:10.1109/ICCCNT.2018.8493915.
65. Hoque, O.B.; Jubair, M.I.; Islam, M.S.; Akash, A.F.; Paulson, A.S. Real time bangladeshi sign language detection using faster r-cnn. In Proceedings of the 2018 International Conference on Innovation in Engineering and Technology (ICIET), Dhaka, Bangladesh, 27–28 December 2018; pp. 1–6.