applied sciences

MDPI

Article

Borno-Net: A Real-Time Bengali Sign-Character Detection and Sentence Generation System Using Quantized Yolov4-Tiny and LSTMs

Nasima Begum *, Rashik Rahman, Nusrat Jahan, Saqib Sizan Khan, Tanjina Helaly *, Ashraful Haque and Nipa Khatun

Department of Computer Science and Engineering, University of Asia Pacific, Dhaka 1205, Bangladesh
* Correspondence: mail4nasima@gmail.com or nasima.cse@uap-bd.edu (N.B.); tanjina@uap-bd.edu (T.H.)

Abstract: Sign language is the most commonly used form of communication for persons with disabilities who have hearing or speech difficulties. However, persons without hearing impairment cannot understand these signs in many cases. As a consequence, persons with disabilities experience difficulties while expressing their emotions or needs. Thus, a sign character detection and text generation system is necessary to mitigate this issue. In this paper, we propose an end-to-end system that can detect Bengali sign characters from input images or video frames and generate meaningful sentences. The proposed system consists of two phases. In the first phase, a quantization technique for the YoloV4-Tiny detection model is proposed for detecting 49 different sign characters, including 36 Bengali alphabet characters, 10 numeric characters, and 3 special characters. Here, the detection model localizes hand signs and predicts the corresponding character. The second phase generates text from the predicted characters by a detection model. The Long Short-Term Memory (LSTM) model is utilized to generate meaningful text from the character signs detected in the previous phase. To train the proposed system, the BdSL 49 dataset is used, which has approximately 14,745 images of 49 different classes. The proposed quantized YoloV4-Tiny model achieves a mAP of 99.7%, and the proposed language model achieves an overall accuracy of 99.12%. In addition, performance analysis among YoloV4, YoloV4 Tiny, and YoloV7 models is provided in this research.

Keywords: deep learning; natural language processing; sign character detection; sign language; computer vision; Yolov4-Tiny; long short-term memory

Citation: Begum, N.; Rahman, R.; Jahan, N.; Khan, S.S.; Helaly, T.; Haque, A.; Khatun, N. Borno-Net: A Real-Time Bengali Sign-Character Detection and Sentence Generation System Using Quantized Yolov4-Tiny and LSTMs. *Appl. Sci.* **2023**, *13*, 5219. https://doi.org/10.3390/app13095219

Received: 9 March 2023
Revised: 1 April 2023
Accepted: 5 April 2023
Published: 22 April 2023

1. Introduction

The most-widespread communication means for persons with hearing or speech disabilities is sign language. Bengali Sign Language (BSL) is composed of precise hand motions produced with one or two hands, where the arrangement of the fingers and the position of the palm together form the characters of the language [1,2]. The signs that are used to represent Bengali characters and numerals are called BSL. In Bangladesh, there are over 1.7 million people who have hearing impairment or speech impairment. These hearing- or speech-impaired individuals use BSL for communication. However, due to the lack of knowledge of sign language, persons with such disabilities cannot convey their opinions and emotions properly. To address this issue, an interpreter system is needed that can detect Bengali signs characters and translate them into Bengali text. It can reduce the communication gap between person with and without disabilities.

Currently, most research emphasizes vision-based [3] or sensor-based [4] systems. However, further research needs to be done in this domain to introduce an efficient and convenient system for sign-language interpretation. At present, various benchmarking datasets for BdSL detection can be found, and there has been much research on the subject. However, these datasets are insufficient for developing and testing deep learning models,

and most of them are closed-source. As a result, it becomes difficult to train Convolutional Neural Network (CNN) models to detect BSL.

Sign-language detection is one of the challenging tasks in computer vision. Numerous related works [5–9] on detecting the individual characters of Bangla Sign Language can be found. However, hardly any work has been done with generating text from sign images. Many research studies only included around 25–35 sign characters for their detection models. The dataset we have used, has 49 different characters. Additionally, researchers use various techniques, i.e., K-Nearest Neighbors (KNN), Support Vector Machine (SVM), shallow neural networks, and image processing, for detection purposes. However, these models are trained on a small dataset consisting of only 2000–3000 pictures in total. Furthermore, previous works not only used a smaller amount of data but also a smaller number of classes. The maximum number of classes was 36, which was proposed by Hoque et al. [10]. The YoloV4 [11] detection model is quite popular among researchers for detecting sign language [12,13]. Researchers [13] developed a BSL detection model using Yolov4. In addition to detecting or localizing the region of the waved sign in an image or video frame, the YoloV4 model also predicts the class or label of which sign is being waved. In a nutshell, this detection model also recognizes the localized area of the hand sign.

Researchers have utilized the YoloV4 model to detect sign language, which is a popular technique in detection systems. However, this research focuses on optimizing the YoloV4-Tiny model by using quantization techniques. The output labels of the proposed detection model are passed to the LSTM-based text generation model for generating meaningful words from the resultant characters. As far as current literature is concerned, there appears to be a dearth of studies pertaining to the detection of Bangla hand signs through implementation of a YoloV4-Tiny model, and no quantization technique has been put forth in this regard. In this work, we propose an end-to-end system that can detect Bengali sign characters and generate coherent phrases through offering a novel quantization method for the YoloV4-Tiny model (for detecting Bangla hand signs) and LSTM model (for generating meaningful sentences from detected Bangla hand signs). A BdSL 49 dataset consisting of 49 distinct characters, including 36 alphabet characters, 10 numeric characters, and 3 special characters, is used to train the detection model. The dataset has a total of 14,745 images for 49 different characters.

The main contributions of this research are as follows:

- An end-to-end system is proposed for detecting Bengali sign characters and generating meaningful sentences.
- A quantization technique for the YoloV4-Tiny model is proposed for detecting hand signs and predicting the characters and to make it implementable on edge devices.
- A sentence-generation model based on LSTM is proposed that takes input from the predicted characters of the detection model and generates meaningful sentences.
- The proposed system achieves a mAP of 99.7% for the detection phase, as well as an accuracy of 99.12%, which is obtained by the sentence-generation model.
- A comprehensive performance analysis of some detection models such as Yolov4, Yolov4-Tiny, and Yolov7 is also provided.

The rest of the paper is organized in the following manner: Section 2 describes the literature review. Section 3 represents the dataset description. The proposed methodology is described in Section 4. Section 5 illustrates the experimental results analysis. Lastly, Section 6 concludes the paper.

2. Literature Review

Sign Language Recognition (SLR) is a significant computer vision task that has been extensively studied. The prevalent SLR approaches can be categorized into two groups: recognizing a single sign at a time [14] and recognizing multiple signs from video content input [15]. However, there is lack of sufficient study for Bangla SLR and/or detection. This section discusses previous work related to Bangla SLR and detection.

Venugopalan et al. [16] proposed a dataset of commonly used Indian Sign Language (ISL) words for hearing-impaired COVID patients in emergency cases. They also presented a deep convolutional LSTM hybrid model for recognizing the proposed ISL words and acquired an accuracy of 83.36%. The authors in [17] developed a system that can recognize Arabic hand signs of 31 classes and translates them into Arabic speech. The system provides 90% for recognizing Arabic hand signs. The authors propose a deep multi-layered CNN model in [8] to detect 36 static English signs, including digits and alphabet characters and 26 dynamic English signs such as emotional hand gestures. Their proposed model achieved 99.89% accuracy in the blind testing phase. One drawback of this model is that it cannot detect signs in real time. Khan et al. [18] proposed a grammar-based translation model to translate English sentences into equivalent Pakistani Sign Language and got a Bilingual Evaluation Understudy (BLEU) score of 0.78. However, the model is unable to translate compound and complex sentences correctly.

For Bangla, OkkhorNama is an image dataset is proposed in [19] for real-time object detection. It contains over 12,000 images of 46 classes. The dataset is used to train different versions of YOLOv5 for performance analysis. However, the number of classes is not significant. Afterwards, a dataset named BdSL36, which includes 26,713 images of 36 different classes, was proposed by [10]. This dataset is further divided into two sections. The first section contains images for developing a detection model, and the second section contains images for developing a recognition model. To evaluate their dataset, the authors trained ResNet-50 on their dataset and achieved an accuracy of 98.6%. Although the number of images in the dataset is significantly large due to data augmentation, background removal, affine transformation, and many other methods applied, it only has 36 classes. The authors of [20] developed a dataset named Shongket, which consists of 36 classes of distinct Bengali signs. They also trained a few CNN models using their dataset and achieved about 95% accuracy. The dataset BDSLInfinite, consisting of 37 different signs, was developed by Urmee et al. [6] and contains 2000 images, and a trained recognition model using Xceptoin architecture achieved 98.93% accuracy as well as 48.53ms response time.

The authors of [21] proposed a custom CNN architecture and trained it on a dataset that consists of 100 different static sign classes. On colored and grayscale images, the proposed method obtained training accuracy rates of 99.72% and 99.90%, respectively. Basnin et al. [22] proposed an integrated CNN-LSTM model for Bangla Lexical Sign Language Recognition using a dataset that contains 13,400 images of 36 distinct classes of Bangla lexical signs. Their proposed model achieved 90% training accuracy and 88.5% testing accuracy. Moreover, they used high computational techniques for data pre-processing, which made their model computationally expensive.

An application that speaks the results in Bangla after automatically detecting hand-sign-based digits was developed by the authors of [23]. They utilized deep neural networks to complete the task, and their proposed model achieved an accuracy of 92%. Islam et al. [24] used four different finely tuned transfer learning methods to recognize 11 different Bengali sign words and got an accuracy of 98%. Shurid et al. [25] proposed a Bangla sign language recognition and sentence generation model and achieved 90% accuracy using their augmented dataset. However, their proposed model could not work correctly to recognize critical sign gestures. Angona et al. [26] developed a computer system to recognize 36 different classes of BSL and translated them into text format. For sign recognition, they used the MobileNet version 1 model and got an accuracy of 95.71%. Podder et al. [27] designed a sign language transformer that can aid in establishing communication with doctors remotely by translating sign language into text and speech. It also translates the speeches of doctors into sign language. However, the system is not yet tested for people with disabilities in Bangladesh. A Scale-Invariant Feature Transform (SIFT) technique and Convolutional Neural Network (CNN) are used in the proposed system of [7] to recognize one-handed gestures of 38 Bangla Signs. However, the used dataset is relatively low. Rafiq et al. [9] proposed a translation system that can translate different Bengali word signs into Bengali speech using a seven-layered custom sequential CNN

model. The system trained with 1500 images of 10 different word signs and achieved 97% test accuracy as well as an average response time of 120.6ms.

The authors in [5] proposed a method to detect and recognize American Sign Language using the YOLOv5 algorithm and achieved an overall F1-score of 0.98. However, their dataset contains only 36 different sign classes. The authors of [13] proposed a real-time Bangla sign-language detection model and generated textual and audio speech. They created a dataset of 49 different classes containing 12,500 images of BDSL. They used Yolov4 for detecting hand gestures and got an overall accuracy of 97.95%. However, while generating some common words, the system faces some difficulties. A rule-based system is proposed by the authors of [28] to interpret Bengali text and voice speech into BSL. The system achieved an accuracy of 96.03% for voice interpretation and 100% for text translation. However, the system was trained and tested with only Bangla numerals. Khan et al. [29] proposed a CNN and customized Region-of-Interest (ROI) segmentation-based BDSL translator device that can translate only five sign gestures. It demonstrates about 94% accuracy in detecting signs in real-time. Das et al. [30] proposed a hybrid model for automatic recognition of Bangla Sign Language (BSL) numerals and alphabet characters using a deep transfer learning-based convolutional neural network with a random forest classifier. The proposed system is verified on 'IsharaBochon' and 'Ishara-Lipi' datasets and achieves 91.67% and 97.33% accuracy, respectively, for both character and digit recognition. The authors of [31] proposed a method that includes three steps: segmentation, augmentation, and CNN-based classification, and is evaluated on three benchmark datasets. The segmentation approach accurately identifies gesture signs using a concatenated approach with YCbCr, HSV, and a watershed algorithm. A CNN-based model called BenSignNet is applied to extract features and for classification. Hassan et al. [32] present a low-cost Bangla sign language recognition model that uses neural networks to convert signs into Bangla text. The dataset was manually captured, and image processing techniques were used to map actions to relevant text. The system achieves an accuracy of approximately 92% and can be used for applications such as sign language tutorials or dictionaries. Akash et al. [33] proposed a real-time Bangla Sign Language (BdSL) detection system that aims to generate Bangla sentences from a sequence of images or a video feed. The system uses the Blazepose algorithm to detect sign-language body postures and a Long Short-Term Memory (LSTM) network to train the data. After training for 85 epochs, the model achieved a training accuracy of 93.85% and a validation accuracy of 87.14%. The authors of [34] are working on isolated sign language recognition using PyTorch and YOLOv5 for video classification. The system aims to help people with hearing and speech disabilities to interact with society. The authors achieved an accuracy rate of 76.29% on the training dataset and 51.44% on the testing dataset. Table 1 illustrates the overview summary of the literature review section.

After a thorough investigation of the literature, we discovered that there are no works that incorporate TinyML or model optimization in BSL detection or recognition. Additionally, there is no work related to quantizing the Yolov4-Tiny model, and very few works implement sentence generation in their research.

Table 1. Literature overview summary.

Research	Year	Dataset	Method	Performance
Talukder et al. [19]	2021	OkkhorNama	Yolo V5	mAP: 98.02%
Hasan et al. [20]	2021	Shongket	KNN, SVM, Random Forest	Digit Accuracy: 95% Letter Accuracy: 91.3%
Rafiq et al. [9]	2021	Bangla 10 Words Dataset	Custom CNN Model	Test Accuracy: 97%
Talukder et al. [13]	2020	BdSL Dataset	Yolo V4	Overall Accuracy: 97.95%
Angona et al. [26]	2020	BdSL36	MobileNet	Overall Accuracy: 95.71%
Das et al. [30]	2023	IsharaBochon and Ishara-Lipi	Hybrid Model	Overall Accuracy: 91.67%
Miah et al. [31]	2022	BdSL Alphabet, KU-BdSL and Ishara-Lipi	BenSignNet	Overall Accuracy: 94%, 99.60% and 99.60%
Hassan et al. [32]	2022	Author-Developed Private Dataset	Custom CNN Model	Overall Accuracy: 92%
Akash et al. [33]	2023	Author-Developed Private Dataset	Blazepose and LSTM	Overall Accuracy: 87.14%
Tazalli et al. [34]	2022	Author-Developed Private Dataset	Yolo V5	Overall Accuracy: 51.44%

3. Dataset Description

In this research, the dataset BdSL 49 [35] is used. The dataset contains a total of 14,745 images. It has 49 distinct classes, such as 36 Bengali alphabet characters, 10 numeric characters, and 3 special characters (Space, Compound Character, and End of Sentence). The Label and Class name of each character is presented in Figure 1.

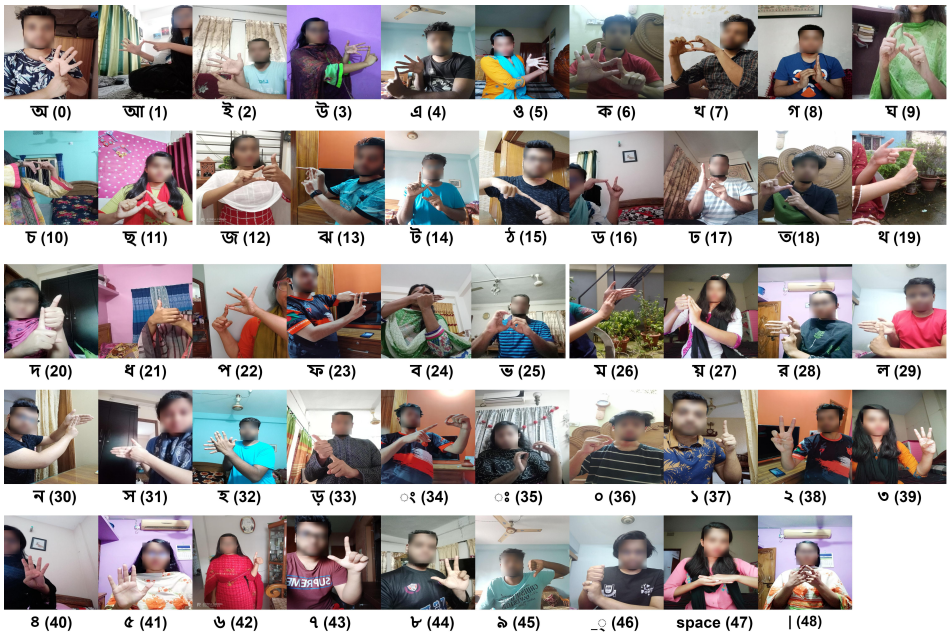

Figure 1. Sample dataset.

Each character has around 300 sample images that are distributed between two phases, which is illustrated in Figure 2. For training purposes, 80% of the samples are used. The remaining 20% of the images are used for testing purposes. The dataset contains images in the RGB format along with a pixel size of 128 × 128.

Figure 2. Data distribution of classes.

There are three special sign, and their purposes are defined below:

1. Compound Character (Label 46): It is known as hasantha and is utilized to create a compound character. The structure of a Compound Character is shown in Figure 3.

Figure 3. Example of Compound Character.

2. Space (Label 47): It is used to create a gap between two words.
3. End of Sentence (Label 48): It is used to indicate a sentence's end.

4. Proposed Methodology

In this paper, we propose a system that can detect Bengali sign characters and numbers and can thus generate meaningful sentences from the detected sign characters. The overall architecture of the proposed system is illustrated in Figure 4. The system is divided into two segments: Sign-Character Detection (SCD) and Sentence Generation (SG). In the first phase, the input images or frames of video are passed to the detection model. Here, the YoloV4-Tiny detection model is quantized and hyperparameter-tuned to localize hand signs and predict characters of the corresponding signs. In the second phase, the encoder–decoder LSTM model is utilized to generate meaningful sentences from the detected sign characters.

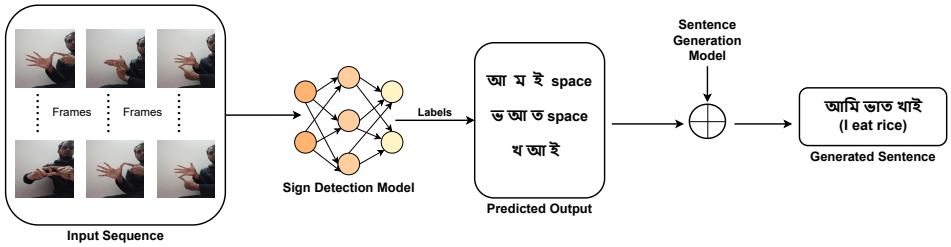

Figure 4. Overview of the proposed end-to-end system.

4.1. Sign-Character Detection

Object detection is a computer vision technique for locating objects in an image or video. The initial part of this study is to identify the coordinates of the Bengali character signs of the given image. For detection purposes, the YOLOv4-Tiny model is used. The

compact version of YOLOv4 is called YOLOv4-Tiny. It is commonly used to detect objects. It is developed to train detection models on low-cost embedded systems such as IoT devices. Furthermore, after training, the YoloV4-Tiny model is quantized to make it more lightweight and faster and to make it implementable on edge devices. YOLOv4-Tiny makes a few modifications to the original YOLOv4 network. To begin with, the number of convolutional layers in the Cross Stage Partial (CSP) backbone is reduced. Furthermore, the number of YOLO layers is decreased from three to two, and there are few anchor boxes for prediction. The model is trained from 29 pre-trained convolutional layers, while YOLOv4 is trained from 137 pre-trained convolutional layers.

Manual tuning of the hyperparameters provides the optimal outcome. Table 2 contains details on these parameters. Here, the momentum equals 0.95 for enhancing the gradient's stability. We set decay to 0.0005 to further stabilize the model. The model's steps are set at (24,000, 27,000). As the model detects only character signs, the value 49 is assigned to the class parameter. The filter parameter for the last convolutional layer is set to 162. Cut Mix is assigned 0 in order to conduct a detection operation. The mosaic and random functions are set to 1. The mosaic function creates a mosaic of four images, and the random function dynamically resizes the images between 1/1.4 and 1.4 for each of the ten iterations, which creates a model with a significant level of generalizability. In the final layer, both mosaic and random data augmentation functions are utilized. The batch size is configured to 64 and the subdivision is set to 16 for the model; as a consequence, we may send 64/16 = 4 images into the model in each mini-batch for quicker convergence. The model is trained for 30,000 epochs.

Table 2. Final hyperparameters.

Hyperparameter	Value	Hyperparameter	Value
momentum	0.95	classes	49
decay	0.0005	Cut_Mix	0
steps	24,000, 27,000	random	1
burn_in	1000	mosaic	1
batch size	64	subdivision	16
channel	3	filter	162

Figure 5 illustrates that YoloV4-Tiny has three fundamental components, which are (i) CSPDarknet53-Tiny, (ii) a Feature Pyramid Network (FPN), and (iii) a YOLO Head. For main feature extraction, CSPDarknet53-Tiny is used, which is composed of a convolutional block (Conv) and a CSPBlock. Batch normalization and activation functions are included in convolutional layers.

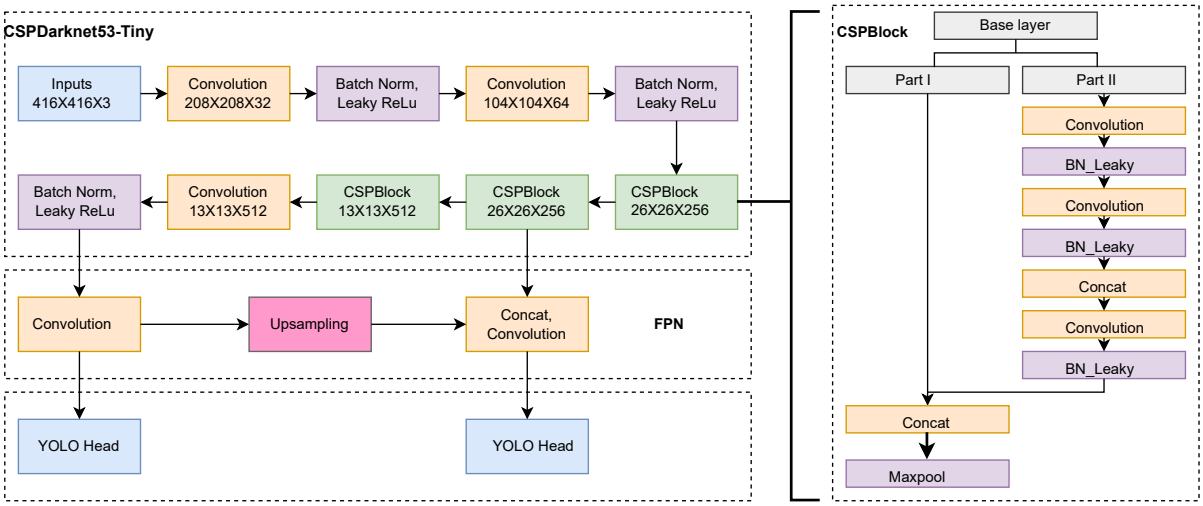

Figure 5. Overall architecture of sign-detection model (YOLOv4-Tiny).

The model is regulated through batch normalization. This replaces the necessity of using dropout layers in the design to avoid overfitting issues. By specifying the variance values, the model improves the normalization of its input. The LeakyReLU activation function in Equation (1) is used in YOLOv4-Tiny, where a_i is a constant parameter.

$$y_i = \begin{cases} x_i & x_i \geq 0 \\ \frac{x_i}{a_i} & x_i < 0 \end{cases} \quad (1)$$

Cross Stage Partial Block (CSPBlock) divides the Base layer model in half based on the features of Cross Stage Partial Networks (CSPNet). After a series of convolutional operations, the first component is created as a residual edge and the second portion is merged with the previous to generate the final output. The Feature Pyramid Network (FPN) structure may incorporate characteristics from several network levels, including contextual information from deep networks and geometrical information from low-level networks. Consequently, this enhances the capacity to extract features. The YOLO Head is the final feature-output module in the design. In the framework of a single-stage detector, the YOLO Head is responsible for performing dense prediction.The final output is a dense prediction, which consists of a vector containing the coordinates (U_x, U_y, U_w, U_h) of the predicted bounding boxes as shown in Equation (2), where Q_w and Q_h denote the width and height of the bounding boxes, respectively, σ represents the sigmoid function, and e represents the exponential. The values (V_x, V_y) denote the top-left corner of the image's coordination, P_x is the predicted x-coordinate offset, and P_y is the predicted y-coordinate offset; t_w and t_h are the predicted bounding box width and height offset, respectively.

$$\begin{aligned} U_x &= \sigma(P_x) + V_x, \\ U_y &= \sigma(P_y) + V_y, \\ U_w &= Q_w \cdot e^{t_w}, \\ U_h &= Q_h \cdot e^{t_h} \end{aligned} \quad (2)$$

The YOLOv4-Tiny model optimizes the loss function through parameter iterative regression to achieve the model optimization objective. The total loss function has three components: Classification (CLF) loss, Confidence (CNF) loss, and Complete Intersection Over Union (CIOU) loss. The total loss is represented in Equation (3).

$$TotalLoss = CLFLoss + CNFLoss + CIOULoss \quad (3)$$

In order to figure out whether the target is included, confidence loss is utilized. Classification loss is utilized to fit the positioning information, which consists of aspect ratio, overlap area, and center point distance. CIOU loss is the combination of the overlap area, the distance, and the aspect ratio.

4.2. YoloV4-Tiny Model Quantization

This section describes the proposed quantization of the YoloV4-Tiny model. Quantization for deep learning is the modeling of a neural network with a small bit width that receives floating-point input. Quantization algorithms have gained prominence in deep-learning model-optimization strategies for overcoming resource limitation difficulties and reducing the computational burden of IoT devices. Quantization strategies employ integer-format or low-precision floating-point operations to shorten inference time and minimize model size for deep neural networks. In addition, hardware can accelerate integer calculations with minimal loss of model fidelity due to quantization. Since it supports computations in the integer domain, such as matrix convolutions and multiplications, the system under consideration emphasizes consistent integer quantization. This allows for the execution of better integer arithmetic operations on end devices with less RAM, computing capacity, and flash memory, such as microcontrollers. Using integer operations with a high throughput reduces inference lag.

In Equation (5), q denotes the bit representation of a quantized value, whereas i represents a real number. As it is conventional to train neural nets in floating point and then quantize the generated weights, we adopt integer-only quantization after training. After implementing the rounding behavior of the quantization technique into floating-point values, the following can be observed:

- The values are quantized prior to correlating the weights with the input. When the layer is normalized with batch normalization preparatory to quantization, the batch normalization variables are incorporated into the weights $\hat{e}$ using Equation (4), where λ is the batch-normalization-scale parameter, σ_F^2 is a rolling average approximation of the batch-wide variance of convolution results, e is weights, and ε is a tiny constant
- Each element's quantization is based on the number of convolution layers, the cutting range, and the point-wise quantization coefficient q provided in Equation (5).

$$\hat{e} = \frac{\lambda * e}{(\sigma_F^2 + \varepsilon)^{\frac{1}{2}}} \tag{4}$$

$$clip(i,j,k) = min(max(l,m),a)$$

$$s(j,a,T) = \frac{a-j}{T-1}$$

$$q(i,j,k,T) = \lfloor \frac{clip(i,j,k) - u}{s(j,k,T)} \rceil * s(j,k,T) + j \tag{5}$$

where i represents the real number that needs to be quantized, (j,k) represents the quantization spectral range, T determines the number of quantization stages, and $\lfloor \rceil$ represents rounding to the closest integer. T is set to 8 because for 8-bit quantization $T = 2^8 = 256$ is utilized. Quantization ranges for weight quantization and activation quantization are treated separately. Both situations entail moving the boundaries (j,k). As a result, the learned quantization variables equate to the scale S and zero-point Z as specified in Equation (6).

$$i = S(q - Z) \tag{6}$$

4.3. Sentence Generation

The encoder–decoder architecture is utilized to build sequence-to-sequence (seq2seq) models. The Seq2seq model is a special class of RNN architecture that is used to solve sequential language-related problems. In this research, we used a sequence-to-sequence LSTM (Long Short-Term Memory) model, which is also known as encoder–decoder LSTM. This model is used to build meaningful sentences from a sequence of inputs. There are two main components in the proposed LSTM models: Encoder LSTM and Decoder LSTM. The summary of the proposed LSTM model is presented in Figure 6.

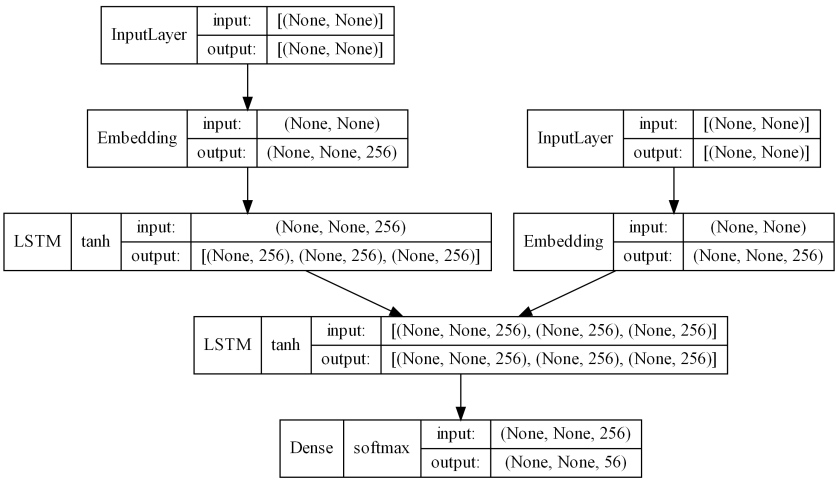

Figure 6. Architecture of the proposed sentence-generation model (LSTM).

4.3.1. Encoder LSTM

In the encoder LSTM, the model reads the input sequence character-by-character. Then, it stores the internal states h_i and c_i of the LSTM network, where the length of the sentence is assumed as i. The vectors h_i and c_i encode the input sequence in a vector form, which is known as input-sequence encoding.

4.3.2. Decoder LSTM

The decoder LSTM plays a slightly different role in the training phase and inference phase. The decoder is initially configured to generate the output sequence based on the encoded data. The initial states of the decoder (h_0, c_0) are set to the final states of the encoder. In the training phase, a start token is added before each word, and an end token is added to the end of each sentence. These two tokens are used to indicate the starting and ending of the sentence while decoding. The "Teacher Forcing" technique is used to train the decoder. The approach utilizes the output of the preceding time-step as the input for the subsequent time-step of the decoder. Afterward, the loss is estimated on the expected outputs from each input sequence. The input sequence is sent through an embedding layer in both the decoder and the encoder to minimize the lengths of the input character vectors since one-hot-encoded vectors may be relatively big. The embedding layer decreases the size of the word vectors from four to three in the encoder section, as shown below. At the output layer of the proposed model, the softmax activation function in Equation (7) is used to handle multiple classes.

$$\sigma(\bar{w})_x = \frac{e^{w_x}}{\sum_{y=1}^{z} e^{w_y}} \tag{7}$$

where σ represents the softmax function, the input vector is $(\bar{w})$, the standard exponential function for the input vector is e^{w_x} and for the output vector is e^{w_y}, and the number of classes in the multiclass classifier is z .

Lastly, errors are back-propagated throughout the time to update the parameters of the network. During the inference phase, the decoder is called in a loop. The initial input of the decoder is always the start token. The loop breaks when the end token is executed by the decoder. Figure 7 illustrates the overall workflow of the proposed language model.

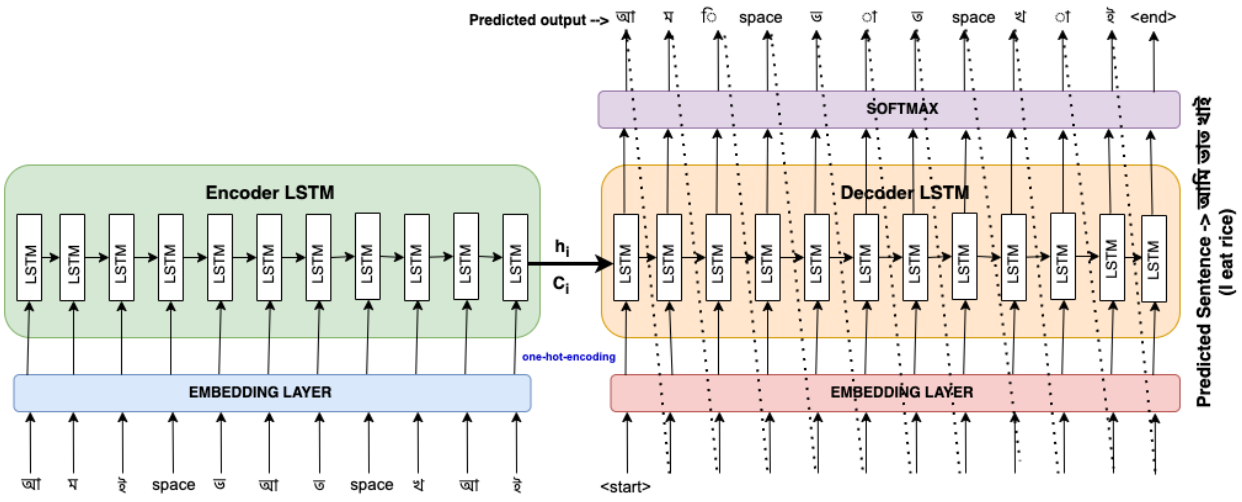

Figure 7. Workflow of the proposed LSTM architecture.

4.3.3. Parameter Settings

As a loss function, the categorical crossentropy is utilized during training. The size of each batch has been set at 128. Furthermore, RMSprop is chosen as the optimizer. For each parameter, the RMSprop optimizer selects a different learning rate , which dramatically

improves model performance. While using the RMSprop optimizer, the model's weights are altered using Equations (8) and (9).

$$v_t = \beta_{t-1} + (1 - \beta) * q_{t^2} \tag{8}$$

$$w_n = w_o - \frac{n}{\sqrt{v_t + \epsilon}} * q_t \tag{9}$$

where the gradient's average movement speed is v_t, the cost is q_t, and the moving parameter is β. In Equation (9), w_n represents the new weight, the old weight is w_o, the learning rate is (η), and ϵ represents a constant value.

5. Experimental Results Analysis

In this section, the training and validation results for the proposed system are presented. Additionally, the predicted outcome of the proposed system is illustrated in Figure 8. Furthermore, we explain our research findings in this section. The end-to-end system's average response time is 50 ms, which indicates superior real-time performance. The average response time of the detection and sentence-generation models is approximately 30 ms and 20 ms, respectively. These results were obtained by averaging 100 trial tests.

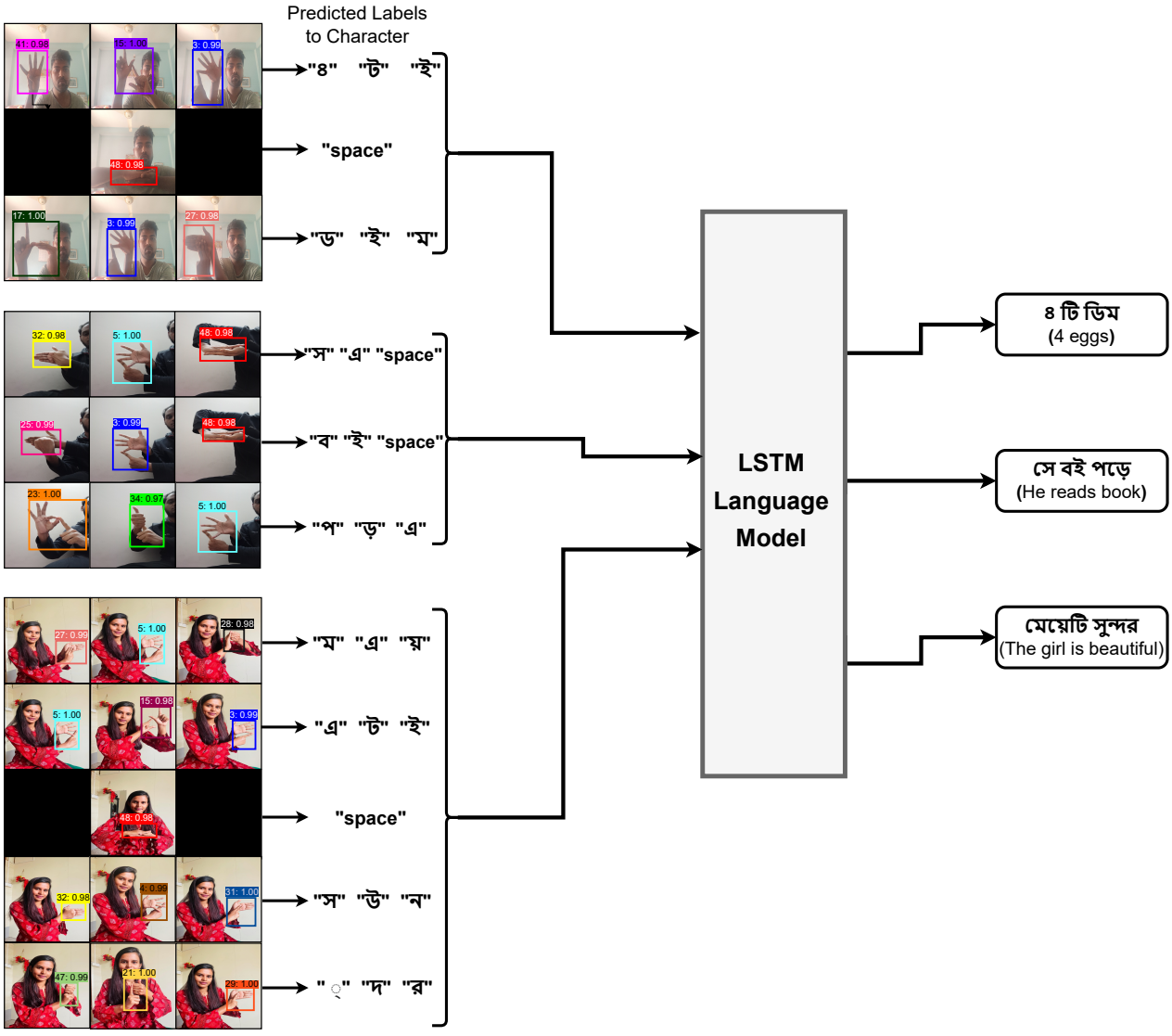

Figure 8. Predicted labels of characters and associated text.

5.1. Evaluation of the YoloV4-Tiny Model

The proposed end-to-end system detects Bengali sign characters and numbers using the quantized Yolov4-Tiny model. Equation (10) is used to calculate the model's mean average precision (mAP) or accuracy, where AP_m represents the average precision of the class m, and the number of total classes is represented by n.

$$mAP = \frac{1}{n} \sum_{m=1}^{m=n} AP_m \tag{10}$$

Figure 9 depicts the loss and mean average precision (mAP) of the proposed model. The model converges after 9000 epochs with a maximum stable accuracy of 99.7% as well as a minimum loss of 0.0816, where the x-axis represents the number of epochs and the y-axis represents the amount of loss. The initial task of the proposed system is to detect Bengali character signs in the given input sequence. The model is trained for 30,000 epochs. Initially, the loss is extremely substantial, but after 1000 epochs it drops below 2.0 and then decreases progressively. After 9000 epochs, the model converges when the loss reaches approximately 0.2. The proposed model achieves accuracy of 99.7%. Figure 8 depicts the detection of a character sign in images captured from a variety of angles and under varied lighting conditions. As a consequence of being trained on a large dataset and having its hyperparameters fine-tuned, the proposed model can detect character signs, which is consistent with the goal of this research.

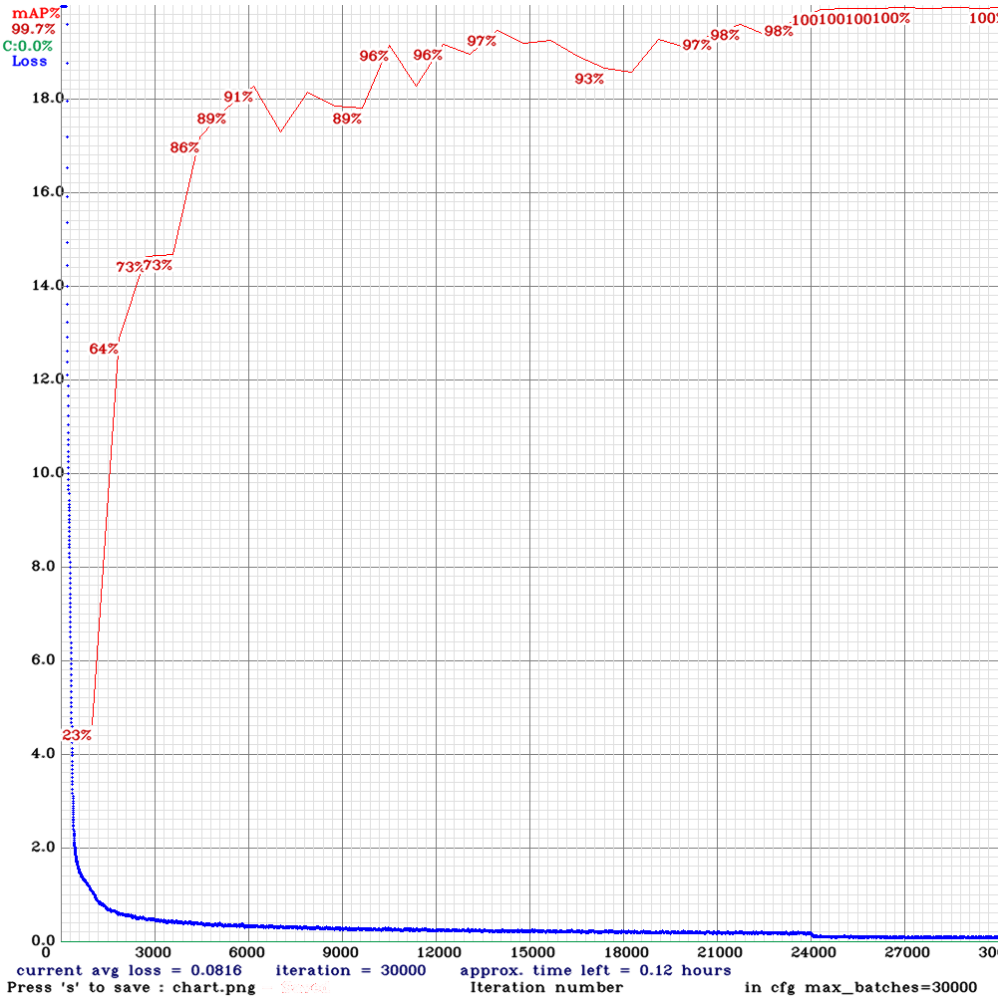

Figure 9. The mAP and loss of the proposed detection model.

5.2. Performance Comparison of Yolo Models

This section describe the comparisons between the proposed Quantized YoloV4-Tiny model with prior YOLO models, which are also trained and evaluated with the BDSL 49 [35] dataset. The models are compared based on their flash occupancy, average mAP, average inference or response time, and average Frames Per Second (FPS) after 100 trials. Additionally, some previous works are recreated to be trained on the BDSL 49 [35] dataset and to evaluate the results. Thus, a comparison of other models to the proposed model is presented in Table 3.

The comparison of previous research endeavors is limited to the BDSL 49 dataset, as other currently accessible datasets lack annotated images. The sole dataset that currently offers annotated images is BDSL 49, which is notable for its comprehensive and equitable distribution of data. From Table 3, it is observed that when the model is trained on the BDSL 49 dataset, the proposed quantized YoloV4-tiny model achieves an accuracy of 99.7%, which outperforms other methods from prior work.

Table 3. Comparison between proposed and other sign-detection models.

Research	Year	Dataset	Detection Model	mAP
Talukder [13]	2020	BDSL 49	YoloV4	96.95%
Hoque [10]	2020	BDSL 49	YoloV3	55.3%
D. Talukder [19]	2021	BDSL 49	YoloV5	97%
Dima [5]	2021	BDSL 49	YoloV5	97.8%
Proposed model	**2023**	**BDSL 49**	**Quantized YoloV4-Tiny**	**99.7 %**

The comparison of detection models is illustrated in Figure 10. YoloV4, YoloV4-Tiny, and YoloV7 were trained and evaluated utilizing the BDSL 49 dataset. After quantization, the quantized YoloV4-Tiny model's results were generated. YoloV4 has the highest flash occupancy, whereas the proposed Quantized YoloV4-Tiny has the lowest of only 12 MB. Concurrently, the proposed Quantized YoloV4-Tiny has the quickest response time of 30 ms. As a result, it can work in real-time with an average FPS of 33. This means that the model can predict 33 frames in one second. The proposed Quantized YoloV4-Tiny has superior real-time performance compared to other Yolo models despite the fact that these models have comparable accuracy, with YoloV7 being marginally less accurate.

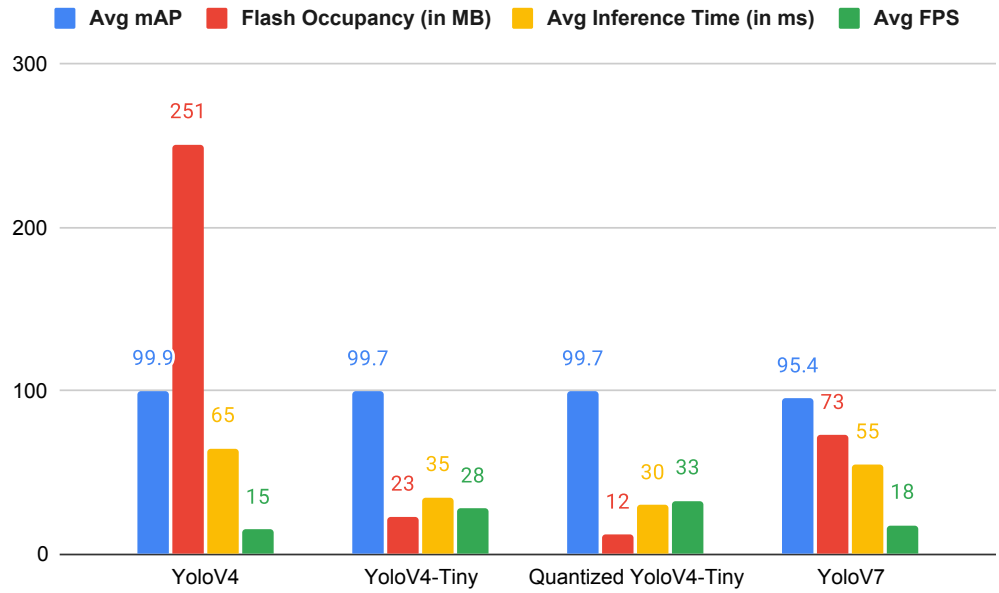

Figure 10. Performance comparison between proposed detection model and other Yolo models.

5.3. Evaluation of Language Model

The encoder–decoder LSTM model was trained for 30 epochs. Figure 11 illustrates the accuracy curve and the loss curve. The proposed language model achieved 99.12% accuracy with a loss of less than 0.02. Here, the blue line illustrates the training score, while the orange line illustrates the test result. This graph demonstrates that the training score converges after the first epoch and then remains constant during the remaining epochs. By contrast, after substantial fluctuations the test score converges within one epoch. Moreover, after convergence, the training and test scores become nearly identical. Despite the fact that the test score is just slightly lower than the training score, we can conclude that the proposed model is highly generalized.

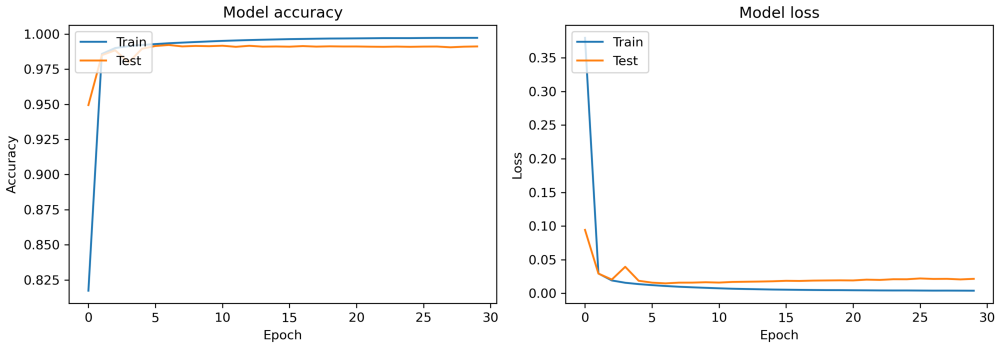

Figure 11. LSTM accuracy and loss curve.

The proposed language model's performance is measured by the BLEU score. BLEU is an acronym for BiLingual Evaluation Understudy. The BLEU score is a statistic for measuring the similarity between a generated sentence and a reference sentence. Perfect matches get a BLEU score of 1, while perfect mismatches receive a BLEU score of 0. The cumulative BLEU score is calculated using Equation (11), where L_p denotes the predicted sentence's length, L_o denotes the length of the original sentence, and P denotes precision. The BLEU score represents the most-reliable metric for measuring a language model's performance. The proposed language model achieved a cumulative BLEU score of 0.93, which is standard.

$$BLEU = MIN(1 - \frac{L_o}{L_p}, 0) + \sum_{n=1}^{4} \frac{logP_n}{4} \quad (11)$$

6. Conclusions

Due to their inability to communicate verbally, people with disabilities are the most-overlooked in society. Additionally, the majority of individuals in society are unable to understand sign language, which results in a communication gap. In order to address this issue, we developed a complete system capable of detecting Bengali sign characters and generating meaningful Bengali sentences. The proposed model was trained on a large dataset, entitled 'BdSL 49', that closely matches real-world circumstances in order to build an extremely accurate model. The dataset contains 49 classes representing the standard sign representations of Bengali Sign Language. In addition, the proposed system is efficient because it requires minimal resources and can operate in real-time on a low-end device or embedded system. Furthermore, the proposed system obtained an accuracy of 99.12% for the proposed language model and 99.7% for the proposed detection model. To lessen the suffering of persons with speech impairment and hard-of-hearing people, we plan to enhance the proposed system to generate speech from video streaming and embed it in an embedded device in the future.

Author Contributions: Conceptualization, N.B., R.R. and T.H.; Data curation, R.R., S.S.K., A.H. and N.K.; Formal analysis, N.B., S.S.K., A.H., N.K. and N.J.; Funding acquisition, N.B. and T.H.; Investigation, N.B., R.R. and T.H; Methodology, N.B., N.J., R.R. and T.H.; Software, N.B., N.J., R.R. and S.S.K.; Supervision, N.B. and T.H.; Validation, N.J., R.R. and T.H.; Visualization, N.B., R.R., S.S.K., A.H., N.K. and N.J.; Writing—original draft, N.B., N.J., S.S.K., A.H., N.K. and R.R.; Writing—review and editing, N.B., N.J., R.R. and T.H. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by the Institute of Energy, Environment, Research, and Development (IEERD), University of Asia Pacific (UAP), Bangladesh.

Institutional Review Board Statement: Not spplicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Dataset is publicly available at the Mendeley Repository: https://data.mendeley.com/datasets/k5yk4j8z8s/5, accessed on 8 March 2023.

Acknowledgments: We give special thanks to the University of Asia Pacific for supporting this research and thanks to IEERD, University of Asia Pacific, for supporting this research project.

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

References

1. Sanzidul Islam, M.; Sultana Sharmin Mousumi, S.; Jessan, N.A.; Shahariar Azad Rabby, A.; Akhter Hossain, S. 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. [CrossRef]
2. Rahaman, M.A.; Jasim, M.; Ali, M.H.; Hasanuzzaman, M. Bangla language modeling algorithm for automatic recognition of hand-sign-spelled Bangla sign language. *Front. Comput. Sci.* **2020**, *14*, 143302. [CrossRef]
3. 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. [CrossRef] [PubMed]
4. 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, India, 26–28 February 2020; IEEE: New York, NY, USA, 2020; pp. 140–144.
5. Dima, T.F.; Ahmed, M.E. Using YOLOv5 Algorithm to Detect and Recognize American Sign Language. In Proceedings of the 2021 International Conference on Information Technology (ICIT), Amman, Jordan, 14–15 July 2021; IEEE: New York, NY, USA, 2021; pp. 603–607.
6. 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; IEEE: New York, NY, USA, 2019; pp. 1–5.
7. 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; IEEE: New York, NY, USA, 2018; pp. 1–6.
8. Bhadra, R.; Kar, S. Sign Language Detection from Hand Gesture Images using Deep Multi-layered Convolution Neural Network. In Proceedings of the 2021 IEEE Second International Conference on Control, Measurement and Instrumentation (CMI), Kolkata, India, 8–10 January 2021; IEEE: New York, NY, USA, 2021; pp. 196–200.
9. Rafiq, R.B.; Hakim, S.A.; Tabashum, T. Real-time Vision-based Bangla Sign Language Detection using Convolutional Neural Network. In Proceedings of the 2021 International Conference on Advances in Computing and Communications (ICACC), Kochi, India, 21–23 October 2021; IEEE: New York, NY, USA, 2021; pp. 1–5.
10. Hoque, O.B.; Jubair, M.I.; Akash, A.F.; Islam, S. Bdsl36: A dataset for bangladeshi sign letters recognition. In Proceedings of the 15th Asian Conference on Computer Vision, Kyoto, Japan, 30 November–4 December 2020.
11. Bochkovskiy, A.; Wang, C.Y.; Liao, H.Y.M. Yolov4: Optimal speed and accuracy of object detection. *arXiv* **2020**, arXiv:2004.10934.
12. Ma, D.; Hirota, K.; Dai, Y.; Jia, Z. *Dynamic Sign Language Recognition Based on Improved Residual-LSTM Network*; IEEE: New York, NY, USA, 2021.
13. Talukder, D.; Jahara, F. Real-time bangla sign language detection with sentence and speech generation. In Proceedings of the 2020 23rd International Conference on Computer and Information Technology (ICCIT), Tejgaon, Dhaka, 19–21 December 2020; IEEE: New York, NY, USA, 2020; pp. 1–6.
14. Wang, H.; Chai, X.; Hong, X.; Zhao, G.; Chen, X. Isolated sign language recognition with grassmann covariance matrices. *ACM Trans. Access. Comput. (TACCESS)* **2016**, *8*, 1–21. [CrossRef]
15. Camgoz, N.C.; Hadfield, S.; Koller, O.; Ney, H.; Bowden, R. Neural sign language translation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 7784–7793.

16. Venugopalan, A.; Reghunadhan, R. Applying Hybrid Deep Neural Network for the Recognition of Sign Language Words Used by the Deaf COVID-19 Patients. *Arab. J. Sci. Eng.* **2022**, *48*, 1349–1362. [CrossRef] [PubMed]
17. Kamruzzaman, M. Arabic sign language recognition and generating Arabic speech using convolutional neural network. *Wirel. Commun. Mob. Comput.* **2020**, *2020*, 3685614. [CrossRef]
18. Khan, N.S.; Abid, A.; Abid, K. A novel natural language processing (NLP)–based machine translation model for English to Pakistan sign language translation. *Cogn. Comput.* **2020**, *12*, 748–765. [CrossRef]
19. Talukder, D.; Jahara, F.; Barua, S.; Haque, M.M. OkkhorNama: BdSL Image Dataset for Real Time Object Detection Algorithms. In Proceedings of the 2021 IEEE Region 10 Symposium (TENSYMP), Jeju, Republic of Korea, 23–25 August 2021; IEEE: New York, NY, USA, 2021; pp. 1–6.
20. Hasan, S.N.; Hasan, M.J.; Alam, K.S. Shongket: A Comprehensive and Multipurpose Dataset for Bangla Sign Language Detection. In Proceedings of the 2021 International Conference on Electronics, Communications and Information Technology (ICECIT), Khulna, Bangladesh, 14–16 September 2021; IEEE: New York, NY, USA, 2021; pp. 1–4.
21. Wadhawan, A.; Kumar, P. Deep learning-based sign language recognition system for static signs. *Neural Comput. Appl.* **2020**, *32*, 7957–7968. [CrossRef]
22. Basnin, N.; Nahar, L.; Hossain, M.S. An integrated CNN-LSTM model for Bangla lexical sign language recognition. In Proceedings of the International Conference on Trends in Computational and Cognitive Engineering, Online, 21–22 October 2021; Springer: Berlin/Heidelberg, Germany, 2021; pp. 695–707.
23. Ahmed, S.; Islam, M.; Hassan, J.; Ahmed, M.U.; Ferdosi, B.J.; Saha, S.; Shopon, M. Hand sign to Bangla speech: A deep learning in vision based system for recognizing hand sign digits and generating Bangla speech. *arXiv* **2019**, arXiv:1901.05613.
24. Islam, M.M.; Uddin, M.R.; AKhtar, M.N.; Alam, K.R. Recognizing multiclass Static Sign Language words for deaf and dumb people of Bangladesh based on transfer learning techniques. *Informatics Med. Unlocked* **2022**, *33*, 101077. [CrossRef]
25. Shurid, S.A.; Amin, K.H.; Mirbahar, M.S.; Karmaker, D.; Mahtab, M.T.; Khan, F.T.; Alam, M.G.R.; Alam, M.A. Bangla Sign Language Recognition and Sentence Building Using Deep Learning. In Proceedings of the 2020 IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE), Gold Coast, Australia, 16–18 December 2020; IEEE: New York, NY, USA, 2020; pp. 1–9.
26. Angona, T.M.; Shaon, A.S.; Niloy, K.T.R.; Karim, T.; Tasnim, Z.; Reza, S.S.; Mahbub, T.N. Automated Bangla sign language translation system for alphabets by means of MobileNet. *TELKOMNIKA (Telecommun. Comput. Electron. Control.)* **2020**, *18*, 1292–1301. [CrossRef]
27. Podder, K.K.; Tabassum, S.; Khan, L.E.; Salam, K.M.A.; Maruf, R.I.; Ahmed, A. Design of a sign language transformer to enable the participation of persons with disabilities in remote healthcare systems for ensuring universal healthcare coverage. In Proceedings of the 2021 IEEE Technology & Engineering Management Conference-Europe (TEMSCON-EUR), Virtual, 17–20 May 2021; IEEE: New York, NY, USA, 2021; pp. 1–6.
28. Rahaman, M.A.; Hossain, M.P.; Rana, M.M.; Rahman, M.A.; Akter, T. A rule based system for bangla voice and text to bangla sign language interpretation. In Proceedings of the 2020 2nd International Conference on Sustainable Technologies for Industry 4.0 (STI), Dhaka, Bangladesh, 19–20 December 2020; IEEE: New York, NY, USA, 2020; pp. 1–6.
29. Khan, S.A.; Joy, A.D.; Asaduzzaman, S.; Hossain, M. An efficient sign language translator device using convolutional neural network and customized ROI segmentation. In Proceedings of the 2019 2nd International Conference on Communication Engineering and Technology (ICCET), Nagoya, Japan, 12–15 April 2019; IEEE: New York, NY, USA, 2019; pp. 152–156.
30. Das, S.; Imtiaz, M.S.; Neom, N.H.; Siddique, N.; Wang, H. A hybrid approach for Bangla sign language recognition using deep transfer learning model with random forest classifier. *Expert Syst. Appl.* **2023**, *213*, 118914. [CrossRef]
31. Miah, A.S.M.; Shin, J.; Hasan, M.A.M.; Rahim, M.A. BenSignNet: Bengali Sign Language Alphabet Recognition Using Concatenated Segmentation and Convolutional Neural Network. *Appl. Sci.* **2022**, *12*, 3933. [CrossRef]
32. Hassan, N. Bangla Sign Language Gesture Recognition System: Using CNN Model. *Sci. Prepr.* **2022**. [CrossRef]
33. Akash, S.K.; Chakraborty, D.; Kaushik, M.M.; Babu, B.S.; Zishan, M.S.R. Action Recognition Based Real-time Bangla Sign Language Detection and Sentence Formation. In Proceedings of the 2023 3rd International Conference on Robotics, Electrical and Signal Processing Techniques (ICREST), Dhaka, Bangladesh, 7–8 January 2023; IEEE: New York. NY, USA, 2023; pp. 311–315.
34. Tazalli, T.; Aunshu, Z.A.; Liya, S.S.; Hossain, M.; Mehjabeen, Z.; Ahmed, M.S.; Hossain, M.I. Computer vision-based Bengali sign language to text generation. In Proceedings of the 2022 IEEE 5th International Conference on Image Processing Applications and Systems (IPAS), Genova, Italy, 5–7 December 2022; IEEE: New York. NY, USA, 2022; pp. 1–6.
35. Hasib, A.; Khan, S.S.; Eva, J.F.; Khatun, M.; Haque, A.; Shahrin, N.; Rahman, R.; Murad, H.; Islam, M.; Hussein, M.R.; et al. BDSL 49: A Comprehensive Dataset of Bangla Sign Language. *arXiv* **2022**, arXiv:2208.06827.