

Computer Vision-Based Bengali Sign Language To Text Generation

by

Tonjih Tazalli
18101176

Sumaya Sadbeen Liya
18101152

Zarin Anan Aunshu
18101183

Magfirah Hossain
18301210

Zareen Mehjabeen
21341059

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science

Department of Computer Science and Engineering
Brac University
January 2022

© 2022. Brac University
All rights reserved.

Declaration

It is hereby declared that

1. The thesis submitted is our own original work while completing degree at Brac University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. We have acknowledged all main sources of help.

Student's Full Name & Signature:

Tonjih Tazalli

Tonjih Tazalli
18101176

Sumaya Sadbeen Liya

Sumaya Sadbeen Liya
18101152

Zarin Anan Aunshy

Zarin Anan Aunshu
18101183

Magfirah Hossain

Magfirah Hossain
18301210

Zareen Mehjabeen

Zareen Mehjabeen
21341059

Approval

The thesis titled “Computer Vision-Based Bengali Sign Language To Text Generation” submitted by

1. Tonjih Tazalli (18101176)
2. Sumaya Sadbeen Liya (18101152)
3. Zarin Anan Aunshu (18101183)
4. Magfirah Hossain (18301210)
5. Zareen Mehjabeen (21341059)

Of Fall, 2021 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science on January 20, 2022.

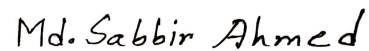
Examining Committee

Supervisor:
(Member)



Dr. Muhammad Iqbal Hossain
Assistant Professor
Department of Computer Science and Engineering
Brac University

Co-Supervisor:
(Member)



Md. Sabbir Ahmed
Lecturer
Department of Computer Science and Engineering
Brac University

Head of Department:
(Chair)

Sadia Hamid Kazi, PhD
Chairperson and Associate Professor
Department of Computer Science and Engineering
Brac University

Abstract

In the whole world, around 7% of people have hearing and speech impairment problems. They use sign language as their communication method. People from various countries use a variety of sign languages. As an example, there are ASL, CSL, JSL, etc. Even in our country, there are lots of people born with hearing and speech impairment problems. So, our primary focus is to work for those people by converting Bangla sign language into text. There are already various projects on Bangla sign language done by other people. However, they focused more on the separate alphabets and numerical numbers. That is why we want to concentrate on Bangla word signs since these people prefer to communicate using words or phrases rather than alphabets. There is not any proper database for Bangla word sign language, so we are making a database for Bangla word sign language for our work. In recognition of sign language (SLR), there usually are two types of scenarios: isolated SLR, which takes words by word and completes recognize action, and the other one is continuous SLR, which completes action by translating the whole sentence at once. We are working on isolated SLR. We introduce a method where we are going to use PyTorch and YOLOv5 for a video classification model to convert Bangla sign language into the text from the video where each video has only one sign language word. Here, we have achieved an accuracy rate of 76.29% on the training dataset and 51.44% on the testing dataset. We are working to build a system that will make it easier for hearing and speech-disabled people to interact with the general public.

Keywords: Bangla Sign Language, Convolutional Neural Network (CNN), Video Classification, PyTorch, YOLOv5, Image Processing.

Dedication

First and foremost, our thesis paper is devoted to all those with hearing and speech impairments.

Then this paper is dedicated to our dear parents and our respected Supervisor, Dr. Muhammad Iqbal Hossain and Co Supervisor, Md. Sabbir Ahmed who has been a constant source of encouragement for us.

Acknowledgement

Firstly, all thanks is attributed to the Almighty, who has graced us all with our well being and has always guided us to the straight path.

Then we will be eternally thankful to our Supervisor, Dr. Muhammad Iqbal Hossain, and Co Supervisor, Md. Sabbir Ahmed, for their unending guidance and assistance whenever we confront an issue.

Finally, we want to express our heartfelt appreciation to our parents, siblings, and friends for their unwavering support throughout our lives.

Table of Contents

Declaration	i
Approval	ii
Examining Committee	iii
Abstract	iv
Dedication	v
Acknowledgment	vi
Table of Contents	vii
List of Figures	ix
List of Tables	xi
Nomenclature	xii
1 Introduction	1
1.1 Thoughts behind the Recognition Model	1
1.2 Aims and Objectives	1
1.2.1 Research Problems	2
1.2.2 Research Objective	2
1.3 About Bangla Sign Language	3
1.4 Bengali Sign Language Recognition	4
2 Related Work	5
2.1 Background Models	8
2.1.1 Convolutional neural network (CNN)	8
2.1.2 Image Processing	8
2.1.3 You Only Look Once(YOLO)	9
2.1.4 PyTorch	10
2.1.5 Video Classification	10
2.2 Table of related works	11
3 Dataset Preparation	13
3.1 Input Data	13
3.1.1 Video Making	14

3.1.2	Video Framing and Frame Selection	15
3.2	Data Preprocessing	15
3.2.1	Data Augmentation	15
3.2.2	Data Labeling	16
4	Proposed Model	18
4.1	Sign Language Detection with Video classification model	19
5	Implementation and Results	21
5.1	Sign Language Detection with Video classification model	21
5.1.1	Training Dataset	21
5.1.2	Classification	23
5.1.3	Sign Detection and Text Generation	24
6	Analysis on Results	26
6.1	Accuracy	26
6.2	Analysis of training and testing sets	30
6.3	Evaluation	31
7	Conclusion and Future Work	33
7.1	Conclusion	33
7.2	Future Work	33
	Bibliography	35

List of Figures

1.1	Some of the vowels and consonants signed by two hands from Bangla sign language dictionary [1].	3
1.2	Some of the basic numerical signs from Bangla sign language dictionary [1].	4
3.1	Flowchart for Dataset creation.	13
3.2	Examples of the diversity of our dataset, which includes a variety of backgrounds, lighting conditions, and signers with varying appearances.	14
3.3	Sign language for word “Classroom”. Only Hand gestures were needed.	15
3.4	Sign language for word “Ghum”. Head movement is also necessary along with hand gestures.	15
3.5	Sign for the word “Baba” Both the right and left hands are used to sign.	16
3.6	Labeling for the word “maa”.	17
3.7	Labeling for the word “Boi”.	17
4.1	Flowchart of the working process.	18
4.2	Variation in YOLOv5 models	19
5.1	Training process.	21
5.2	Mosaics and labels of train batch 0,1 and 2	22
5.3	Model summary of YOLO.	22
5.4	The detection of the word “Bhai” with different accuracy.	23
5.5	The detection of the words “Shahajjo”, “Bhalo”, “Shikkhok”.	23
5.6	Prediction result of classes.	23
5.7	Word Classification after model training.	24
5.8	Different accuracy of the word “Basha”.	24
5.9	Sign detection of the words Dukkrito, Bondhu, Boi, Pakha	25
5.10	Sign detection of the words ghum, bati, classroom, cricket.	25
6.1	Accuracy, Precision, Recall, mAP and loss score charts for train and test sets.	26
6.2	Train Accuracy Bar Chart	28
6.3	Comparison of highest accuracy of all 4 train instances.	28
6.4	Test Highest Accuracy Bar Chart.	30
6.5	Comparison between train-test average accuracy.	30
6.6	vertically aligned video.	32
6.7	Video taken through webcam, too close to camera, cannot detect the sign for “shahajjo” and “doctor”.	32

6.8	Video taken through webcam, full sign can be seen and detected perfectly.	32
-----	---	----

List of Tables

2.1	Summary of related works	12
5.1	Dataset Distribution	21
6.1	Train Accuracy Table (4 Instance)	27
6.2	Test Accuracy Table (1 Instance)	29
6.3	Percentage of accuracy and loss in train and test data	31

Nomenclature

The next list describes several symbols & abbreviation that will be later used within the body of the document

ANN Artificial Neural Network

ASL American Sign Language

BdSL Bangladeshi Sign Language

CNN Convolutional Neural Networ

CSL Chinese Sign Language

CSTL Cognitive Science of Teaching and Learning

GCN Graph Convolutional Network

HAN Hierarchical Attention Network

IoU Intersection over Union

LS Latent Space

NMT Neural Machine Translation

ResNet Residual Neural Network

RNN Recurrent Neural Network

SGD Stochastic Gradient Descent

STMC Spatial-Temporal Multi-Cue

YOLO You Only Look Once

Chapter 1

Introduction

1.1 Thoughts behind the Recognition Model

Language is the method of communication among people which plays a very significant role in our daily life. Everyone cannot hear or talk like normal people, so they use sign language. Sign language is used by speech and hearing-impaired people to communicate with non-sign people and in between themselves. Sign language means vision-based communication through bodily movement: hand and finger movement, lip movement, and facial expression. In our society, deaf, dumb, and sign language users are neglected. It is not easy for them to communicate with normal language-speaking people. That is why they cannot keep pace with other people in any field. The recognition of continuous and real-time sign language in BdSL is challenging. Many scientists are working on developing sign language recognition systems to make them more feasible. Sign language is divided into two types: isolated, in which people communicate using single-word gestures, and continuous, in which a series of movements create a meaningful sentence. Again recognition of signs can be done either by using gloves and a sensor or by using some hardware devices like a leap motion sensor or Microsoft Kinect or by image processing from videos or images. But the first two methods are expensive, so we will approach the last method [14].

To recognize signs, we first need to train our algorithm with the dataset so that it can analyze and then classify and identify the signs. We will make our own dataset. Then we will apply video classification to convert images to the text of the normal Bangla language.

1.2 Aims and Objectives

Signs or alphabets have been employed by humans to communicate, open up and share their ideas and feelings with each other since the dawn of time. Different cultures use different gestures to communicate themselves, and sign language is one of them. Again, there are persons who have physical limitations, such as being deaf, mute, or visually impaired, and they must communicate in certain ways, such as by using hand signals. It is difficult for a deaf child to learn language since they are deprived of voices pronouncing meaningful words from birth. So, for making easy communication with this group of people, our main objective is to generate text from particular signs.

1.2.1 Research Problems

Humans have fundamental communication requirements. Sign language is a medium of speech in which body parts are moved to convey messages. In everyday life, people often use signs in addition to verbal language to communicate with one another. In a nutshell, sign language, facial gestures, and verbal language are used by ordinary people to convey emotions or clarify what they want to say to others. However, sign language is the deaf and hearing impaired people's only mode of communication. Furthermore, sign language is not understood by the majority of the world's population. People who do not have any disabilities are therefore unable to learn a language that they do not need [18].

Sign language is a distinct language, comparable to spoken language, with its own grammar and norms. Different countries express sign language in their own unique ways. As a consequence, there is no standard type of sign language in place yet [19], [3].

As a result, this group has a difficult time engaging with people who do not know BdSL. Automatic Sign Language Recognition will help individuals with hearing and voice impairments communicate with the rest of the world (ASLR). The research started with in-depth studies of a variety of distinct methods and procedures involved with sign language interpretation for both Bangla and other sign languages. It's difficult to interpret signs from videos because the meaning of an expression is dictated by a combination of gestures and hand positions. Users of our system, on the other hand, will be able to render gestures, and the system will take video inputs and show the text that results from such gestures [18].

1.2.2 Research Objective

While most people in Bangladesh are unaware of sign language, there are several works on the subject. Numerous publications on Bangla sign language exist, including alphabets and numerical numbers [3]. However, we want to concentrate on Bangla individual signs since people with disabilities prefer to use words or phrases rather than alphabets to communicate for potential work [13]. Considering that,

1. We will generate a dataset of Bangla sign words using videos and train it with a neural network algorithm.
2. Our approach would take video inputs, fragment them, and extract meaningful features using our proposed method.
3. We will then use our own dataset to train the model and then use the model, software, and algorithms to classify it.

Therefore, as a result, a system will be developed that will make it easier for disabled persons to interact with the general public.

1.3 About Bangla Sign Language

Sign language is a collection of gestures that can interpret information from one to another. Around 7% of the world's population speaks SL as their first language. About 8% of the population in Bangladesh is deaf or mute and Bangla sign language (BdSL) is used by them for communication [12].

Though the development of sign language in Western civilization began in the 17th century as a visual language, Bangladesh lacks appropriate possibilities for learning sign language with almost 30 lakh hearing-impaired people. In 1994, the Ministry of Social Welfare and the Bangladesh National Federation of the Deaf collaborated to create a Bangla sign language dictionary in order to make sign language more accessible to everyone. It is considered as a language of Bangladesh by Prime Minister Sheikh Hasina in 2009, it has no legal protection and no actions have been done to institutionalize the language [2].

As sign language is needed worldwide, different sign languages are used, for example – Arabian, American(ASL), Parisian, Chinese(CSL), etc. Like any other normal language, sign language also has some rules. The structure of Bangla Sign Language (BdSL) differs from that of other sign languages based on phonetics or basic units of SL, and it is based on our country's cultural, social, historical, and religious traditions. BdSL is a modified form of British, American, and Australian sign language, as well as several indigenous signs from around the world [4].

Bangla sign language dictionary employs 36 two-handed Bangla sign alphabets: 6 vowels and 30 consonants and 10 basic numerals shown in figure 1.1 and 1.2 [1].

According to the Bangla Sign Language manual, there are over 5000 sets of signs for the alphabet, numbers, and basic words that are used to communicate by 30 lakh speech and/or hearing-impaired persons [13].

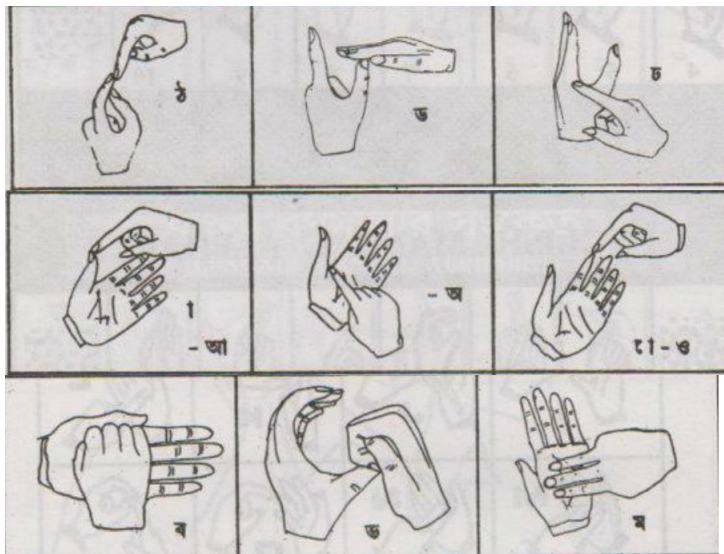


Figure 1.1: Some of the vowels and consonants signed by two hands from Bangla sign language dictionary [1].

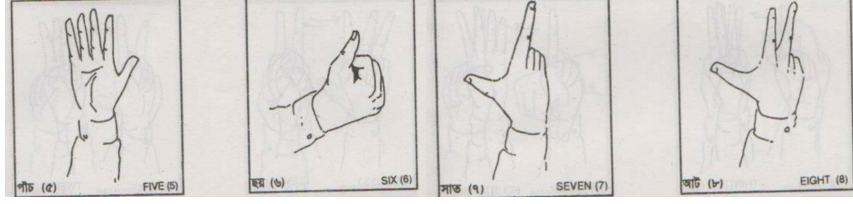


Figure 1.2: Some of the basic numerical signs from Bangla sign language dictionary [1].

1.4 Bengali Sign Language Recognition

Sign Language recognition has recently gained popularity. There are numerous works on sign language in Western nations, the most of them are on ASL (American Sign Language). Despite the fact that there has been little study on Bangla Sign Language Recognition, there is currently no full Bangla Sign Language Recognition system. As a result, there is still a lot of room for advancement in the field of Bangla Sign language recognition study. The reason for this is that most BdSLR systems were designed to recognize just sign alphabets, single characters and digits. However, there are few studies on sign words and phrases, which are essential for communicating.

The sentence structure Bangla sign languages must be done utilizing a word sequence, hand forms, directions, motions, orientations, and placements relating to the human body. In this context, a dynamic sign is a moving gesture defined by a series of hand motions and formations, which may be complemented by body and face expression. As a result, implementing a recognition system for word or sentence level sign language is challenging. For SLR, there are several works on various algorithms and models. These are described in the section on literature reviews.

Chapter 2

Related Work

A large-scale dataset was focused that contains a total of 2000 words that are signed by more than 100 signers [16]. They worked on American Sign Language. They worked on processing and comparing two distinct models: 2D human pose-based approach and holistic visual appearance-based approach. Their findings show that both models achieve relatable results on their dataset, with top-10 accuracy of up to 62.63 percent. A CNN and an RNN construct their first baseline in a visual appearance-based approach to extract the Spatio-temporal features from their given dataset that takes input from video frames. They extracted spatial features using VGG16 pre-trained on ImageNet and then fed the found features to a GRU that is stacked type. In pose based approach, they mainly utilized RNN.

Most of the isolated sensors are the basic building elements of SLR models, with data preprocessing and postprocessing layers on top of that [6]. Temporal segmentation is a difficult task that inevitably leads to errors in subsequent steps. As far as we know, all isolated SLR models need manual tagging of each word in a sentence, which severely limits the amount of data that can be used in training. Instead of segmentation preprocessing, they employed a Hierarchical Attention Network in conjunction with Latent Space (LS-HAN). Two-stream CNN is used to extract video features, followed by LS and HAN for connecting the semantic gap. The new architecture has three elements.

By utilizing the data from the human skeletal movement, an innovative approach was used that separates spatial and temporal data from the dataset and accomplishes merging in the end [18]. Using a Graph Convolutional Network (GCN) and BERT, their proposed architecture clearly captures the spatial interactions in the video. By modeling each found feature detail separately, GCN and BERT get a wide efficiency rate of present newly formed pose-based.

The structured approach for SLT was expanded and divided into two parts: tokenization and translation [19]. The CSTL method analyzes sign language videos to create sign language glosses that capture the meaning of a variety of different signs, as well as to compare and contrast different Neural Machine Translation (NMT) architectures. They achieved superior performance in an end-to-end system that used an STMC network to extract glosses from videos.

Bangla sign language dataset of own hand gestures was used for sign language recognition [13]. The inputs would be captured using a webcam, with the context removed, and then interpreted by a neural network algorithm, which would produce textual output. They constructed a CNN that is identical to the classification model designed by MNIST. Tensor flow and Keras are also used. Video streaming is used to train this model, which is then converted into text. In this article, they demonstrated about 15 different types of Bangla sign language, including alphabets, numbers, and individual signs.

A smartphone with a minimal range front camera captures signs from videos at a frame rate of around 30 frames per second to process real-time sign language detection for Indian sign language [4]. Segmentation, pre-filtering, and feature extraction applied on video frames generate a space for sign language features, which is then used to train and test an artificial neural network algorithm for classifiers. Adaptive thresholding, as well as morphology, provides a segmentation of hand and head parts, which are used to increase the Sobel edge operator's energy. During characterization, word coordinating with score (WMS) offers the presentation of the recommended strategy that has an usual score of about 90% for ANN and an execution point of 0.5221 seconds. This experiment had an identification rate of around 90%.

A real-time SLR was used for Indian Sign Language (ISL) [10]. OpenCV's skin division highlight is used to distinguish and track ROIs for detecting the signs. After that, a machine learning method known as fuzzy c-means clustering is used to prepare and anticipate hand gestures. Human-computer interaction (HCI), game control and sign language interpretation are just a few examples of the many uses for gesture recognition. Hearing and speech-impaired persons may communicate with normal people much more easily thanks to this technology, which recognizes real-time signals. The accuracy of gesture labeling was found to be 75% in this study. About 40 Indian terms were properly identified.

A medium-recognized sign language was utilized and translated into text and Voice and BdSL was used to create real-time written sentences and voice from a recorded video or a collection of images [17]. Several bounding boxes may be predicted using a neural network and a progressive propagation on full images with YOLOv4 as the object identification model. A picture dataset that has 1250 BdSL photos of distinct classes, Bangla alphabets, numerals, and images are taken from 8 different people. CSPDarknet53 and PANet architecture were also utilized as an architecture for their object detection as a sign language finger-spelling approach. The four evaluation criteria are recall, accuracy, F1-Score, and precision. In general, the model was reported to detect objects with an accuracy of 97-97.5 percent.

A structure was created that by its own identifies numbers based on hand signs and conveys the outcome in Bangla language, wherein the hand signal is identified by the proposed structure, generates some digits and conveys them in a line [9]. They experimented with numerous hyper-parameter combinations to obtain the best outcomes in the shortest amount of time with CNN. The structure is commanded and experimented on a data set that is a mixture of two datasets. Dividing into two

sets, for training and testing purposes accordingly. In their trial stage, they took a total of 3200 pictures, 1200 of which were from UNIP and 2000 of which they created themselves. They made the knowledge set consisting of ten sets for every one of the ten digits with 320 images for every sign. 2 major elements of CNN, The Convolution Layer the Pooling Layer are used. Lastly, 89% accuracy on a take a look at sets wherever the validation loss was 0.37. They stated that their model analyzes hand sign based digits with accuracy over 92% valid data that gives it an extremely reliable model.

The author worked on hand action recognition and body language Recognition[15]. The YOLO algorithm was used for this work. It was stated that YOLO is a very useful algorithm in the field of real-time detection. So they wanted to figure out how YOLO works in case of body language detection and how accurate the results can be. Firstly they built their dataset on hand gestures and then body languages step by step. They used PyTorch YOLO to train the model. They came to the conclusion that YOLO is a very speedy algorithm and works well in both hand recognition and body language detection however it sometimes fails in detecting in extreme cases or if the picture quality is not very good.

The author worked on building a system that works on ASL in real Time [22]. CNN with YOLO was used for this work. As sign language focuses mainly on the actions of hand and fingers they preprocessed their acquired data on getting an utilizable area of hand, finger points, different spots of the hands and accurate gestures. They used sub sampling CNN to memorize the preprocessed data and to recognize the hand gestures. They feed calibrated pictures to CNN. They used YOLO to increase the accuracy as YOLO is known to be a very fast algorithm in picture handling. It also has a high precision for detecting ASL and robustness. Their working process is that they firstly take input through the webcam and feed that input into the YOLO algorithm for preprocessing. This deals with labeling and trained dataset. After this other feature extraction and optimizations are done to find accurate hand actions and feed the final result into CNN to recognize and give output in text formation.

The pre-trained YOLOv3 model is retrained to meet the Indonesian Sign Language Requirement [20]. In the feature extraction, just a Convolutional Layer, batch normalization, and the Leaky ReLU Activation function are used. They amassed a collection of 4,547 pictures including 24 different forms of static sign language on their own. The sigmoid function is fed the item and class scores. The data is preprocessed by scaling it to 416×416 pixels and turning it to grayscale. When employing video data, the system achieves 77.14 percent precision, 93.1 percent recall, 72.97 percent accuracy, and an F1 score of 84.38 percent.

A system starts working by receiving real-time input of a Malaysian sign language [21]. A Kaggle dataset with 3000 photos for each sign language was utilized for this work. There are static signs (single hand) and dynamic signs (hand motions) in the dataset. They used Darknet, an open source framework for the YOLO algorithm. Best weights (7000 iterations) were used to evaluate.

2.1 Background Models

2.1.1 Convolutional neural network (CNN)

A convolutional neural network (CNN) is a form of artificial neural network. It is specially intended to process pixel data, classification, segmentation, and other auto correlated data. A CNN employs technology similar to a multilayer perceptron that has been optimized for low processing needs.

To extract spatial characteristics from input pictures, 2D Convolutional Neural Networks (CNN) are commonly employed. A CNN constructs its initial baseline to collect spatial-temporal characteristics from incoming video frames. According to their article, 3D convolutional networks can construct the representation of each frame and also the temporary link between frames in a hierarchical manner. Carreira et al. use ImageNet to increase the 2D filters of the Inception network, which results in perfectly organized 3D filters. The increased 3D filters are tuned on a fine level to complete the actions on the Kinetics dataset to capture better features from the information in a recorded video [16].

The researchers offer a novel two-stream 3D CNN for producing global-local video characteristics. From the paper we know, Because of the nature of signing videos, a comprehensive video feature depiction necessitates the inclusion of both global and local hand locations/motions. The two-stream CNN is initially trained on an isolated SLR dataset, after all weights are fixed and the final SoftMax and fully-connected layer are removed. This CNN is terminated at the first fully connected layer (as a feature extractor) during the testing phase, and each test video is divided into 16 frame segments with a temporal window sliding method, before being fed into the two-stream 3D CNN. The shortened 3D CNN's 4096-dimensional output is the intended global-local feature representation for this particular 16-frame video clip. As a result, each footage is represented as a series of such 4096-dimensional feature vectors [6].

To train the model on each sort of feature, they employed two types of active networks. They utilized CNN in training the architecture on spatial characteristics. Convolutional neural networks are particularly good at identifying local spatial patterns in data. They are particularly effective at identifying patterns and then categorizing images based on those patterns. ConvNets are built with the assumption that the network's input will be an image. Due to the considerable advantages of CNN in extracting spatial characteristics of a picture, they used the TensorFlow library's Inception-v3 model, which is a deep ConvNet, to extract spatial attributes from the frames of video segments [8].

2.1.2 Image Processing

Image processing is a method that involves applying a few procedures on a photograph in order to improve it or extract a few valuable data from it. It's a form of

sign processing in which the input is a photograph and the output is either another photograph or the features or functions associated with that photograph. Photograph processing is one of the most rapidly evolving technologies nowadays. It's a major engineering and computer science research hub as well.

In the study, they offered image Processing model and used a camera to collect data, which was then identified by a neural network system and translated into text using gestures. Keras was used to train this model using video streaming. The translator can quickly transform a video stream of sign language into text, which approach or procedure we can apply in our study to convert Bangla Sign Language to text. We see in the study that they use their system to take pictures, match them with their data set, and display the results in Bangla. They capture many pictures of different people's hand signals in various motions for this project [13].

There is another research study paper that proposes an image processing-primarily based totally version to be able to interpret Bangla Sign Language. Using green techniques to hit upon the hand efficiently the use of pores and skin detection set of rules and eliminating all noise the use of MATLAB and classify the picture in line with the signal gesture performed. According to the paper, the device will take the dataset snapshots and observe a pores and skin detection set of rules on it and hit upon the pores and skin color pixels from it. Then it is going to make the picture binary. Then from the binary picture, it is going to discover the most enormous blob and subtract all different heritage gadgets and most effectively maintain the hand on it. In the subsequent segment for characteristic extraction and evaluation, they use the bag of capabilities in class classification. It splits the picture grid with the aid of using the grid and takes some picture patches from it. The most powerful capabilities are diagnosed, and K- approach clustering is used for vector quantization. The capabilities or bag of phrases are saved withinside the characteristic vector. After that, a multi-magnificence SVM (Support Vector Machine) classifier has been used for categorizing schooling and checking out sets for evaluation [3].

2.1.3 You Only Look Once(YOLO)

YOLO algorithm is utilized in diverse applications to come across various signals, people, and specific signs by using CNN. The algorithm applies a single neural network to the complete image, and next separates the image into parts and predicts bounding boxes and probabilities for every part that is calculated by the predicted probabilities. YOLO can be trained with complete images and after that it increases the sign detection result.

In one paper they worked on both CNN and Yolo Architecture to identify Indonesian Sign Language by giving video data as input. They retrained the YOLOv3 pre-trained model by using Transfer learning which is useful for the number of channels and classes. They used YOLOv3 to assist predictions on three different scales. They applied a detection layer for three feature maps of unequal sizes where each of them contained a stride of 32, 16, and 8. They claimed that the YOLO worked with accuracy on video as well as image. Again they stated that from one sign to another

recognizing the transition frames on video data helped to improve the misrecognition error [20].

Again in another study they stated that the system shows increased accuracy while using YOLO calculation in 45 frames/second which is quite the speediest with high robustness and precision when detecting the ASL. They mentioned that by recognizing the occupancy of trained images, this system will arrange the input images. It can fulfill the predicted result of protest within bounding boxes [22].

However in a research, they focused on observing how the YOLO algorithm can accurately detect some specific body language gestures and also whether they can utilize it in more depth to effectively find out how a person behaves. The YOLO algorithm generalizes both x and y between 0 and 1 and uses h,w,x,y as the data inputs, and they said there x and y portrayed the generalized center of the bounding box (the box around the specific part of the image used to identify) and w and h meant the width and height relative to the image [15].

2.1.4 PyTorch

PyTorch is a deep learning framework which is used for imperative programming. It advances the path of production deployment from research prototyping. It is recognized as Python's learning library, one of the broadly used Machine learning libraries. It works on CPU as well as GPU as an optimized tensor library to deep learn applications.

PyTorch has been discussed which can do automatic differentiation and stimulate GPUs to work on instant implementation of dynamic tensor computations. When compared to the other libraries for machine learning, it also balances the execution. According to them PyTorch provides an array-based programming model. They claimed that the goal of PyTorch is to maintain the interfaces of the design simple and consistent, ideally with one way of doing things by generally declared design. It also models data loaders, and optimizers not complicated and effective by utilizing standard plotting, data processing tools and debugging. PyTorch library handles the basic machine learning complexities inwardly and unexpected performance cliffs. Researchers can manually execute and control the code by the tools pytorch provides that helps to identify their own execution improvements rather than the result the library provides automatically. They also stated that deep learning workflows are benefitted by internal executions of PyTorch as it executes extra features, observes different circumstances, and follows with performance. The features of Python are available all over the complete design execution. PyTorch balances a strict regulation between its control and data flow (i.e. tensors and the operations performed on them) [11].

2.1.5 Video Classification

Video classification is similar to image classification, in that the algorithm uses feature extractors, such as convolutional neural networks (CNNs), to extract feature

descriptors from a sequence of images and then classify them into categories. Video classification using deep learning provides a means to analyze, classify, and track activity contained in visual data sources, such as a video stream. Video classification has many applications, such as human activity recognition, gesture recognition, anomaly detection, and surveillance. Given the video frames, the aim is to create a label that is related to the video. A strong video level classifier not only delivers correct frame labels, but also best represents the complete video based on the features and annotations of the individual frames.

In the study, they conducted an assessment of current video classification systems such as Multiple Instance Learning (MIL), Conditional Random Field (CRFs), and categorizing based on action and gesture [5].

In the paper, they employed a YOLO (You Only Look Once) based detection and classification technique (YOLOv2) to improve computation and processing performance while also identifying objects in video records effectively. Every class of item for which the classification algorithm has been taught provides a bounding box and an annotation characterizing that type of object. The GPU (Graphics Processing Unit) is used in the YOLO-based detection and classification (YOLOv2) to boost the calculation speed and processes at 40 frames per second [7].

2.2 Table of related works

Author Name	Dataset	Methods	Accuracy	Real-time Detection
Li. et al. [16]	Total of 2000 words, signed by more than 100 signers for ASL	CNN	62.63%	No
Huang. et al. [6]	Two open source continuous SLR datasets are used for CSL and GSL	Two-stream 3d CNN,LS-HAN	82.07%	No
Tunga. et al. [18]	WLASL100, WLASL300	GCN,BERT	80.93%, 88.67%	No
Yin et al. [19]	3000 steps done by 8 models for German SLT	CSTL method, STMC network, NMT architecture	96.60%	No
Roy. et al. [13]	15 different types of Bangla signs including alphabets, numbers, and individual signs.	CNN,MNIST	80%	No
Rao. et al. [4]	18 single shot signs for ISL	ANN	90%	Yes
Mariappan. et al. [10]	40 Indian gestures	A machine learning method, fuzzy c-means clustering	75%	Yes
Talukdar. et al. [17]	1250 images of distinct classes, taken by 8 different people for BdSL	CNN,YOLOv4 model, CSP-Darknet53 and PANet architecture	97-97.5%	Yes
Ahmed et al.[9]	320 images of 10 sets for BdSL	CNN	89%, 92%	No
Steve D. et al.[20]	4,547 pictures in 24 different forms for Indonesian Sign Language	YOLO	72.97%	Yes
Zaaba A. et al. [21]	Kaggle dataset with 3000 images of Malaysian SL	Darknet framework, YOLO algorithm.	63.06%	Yes
Bhavadharshini et al.[22]	500 images of American Sign Language	CNN, YOLO	92.5%	Yes

Table 2.1: Summary of related works

Chapter 3

Dataset Preparation

The entire process of creating a dataset for our Bangla Sign Language Recognition is shown below using the flowchart:

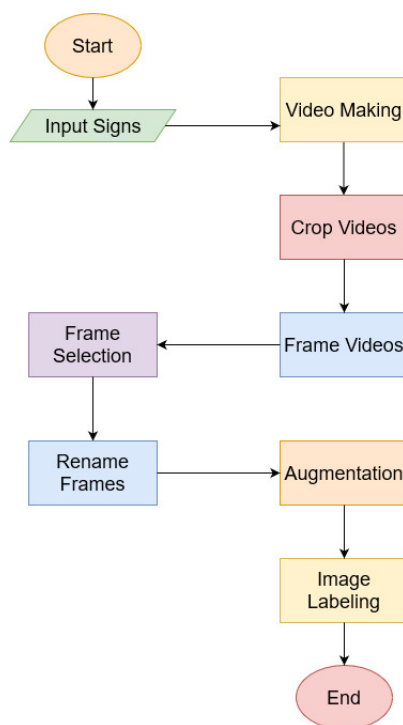


Figure 3.1: Flowchart for Dataset creation.

3.1 Input Data

There are many datasets available for sign language recognition of words in various languages. However, there is no available dataset for Bangla sign language for word detection. That is why we created our own dataset in a similar approach that we read in the paper [16]. We stood in front of the camera and made signs of particular

words and recorded the videos. Each video contained signs of only one word. The videos are done by 5 signers in different backgrounds, lighting, and appearance. To detect the hand movement properly, we cropped the videos accordingly. We applied openCV to convert the videos frame by frame and created a final dataset with selected frames of each word. We created a zip file to upload in the drive so that we can connect the whole dataset with our collab file easily.

3.1.1 Video Making

Firstly, we all recorded videos of us showing sign language of different words. Each video contained only one word where we showed the sign language of one word twice or thrice so that when we frame the videos we get clear frames of the signs. Each video's length is around 4 to 8 seconds. After making the videos we cropped the videos where only our hands are visible. However, all sign languages cannot be shown by only using hands. For sign languages where our head or mouth was necessarily needed, we cropped them where the whole upper body was visible. For new videos we did not crop the video to see how those frames work in our recognition system. We created videos for around 34 words which are used on a daily basis. For example: good, bad, mother, father, morning etc.



Figure 3.2: Examples of the diversity of our dataset, which includes a variety of backgrounds, lighting conditions, and signers with varying appearances.

3.1.2 Video Framing and Frame Selection

We used OpenCV to convert the video into frame by frame pictures. We used python to do the codes. We got around 40 to 90 or more frames for each video where few frames were blurry and hardly recognizable. That is why, after framing all the videos in separate folders we selected 20 frames for each word for our dataset so that the word can be recognized. For one word we made 5 instances in the final dataset done by different people, different backgrounds to create diversity.

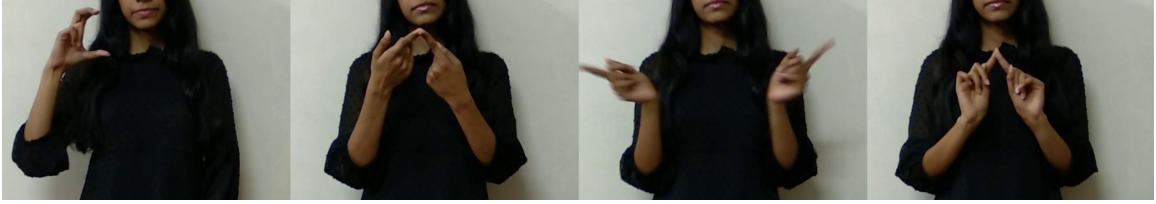


Figure 3.3: Sign language for word “Classroom”. Only Hand gestures were needed.



Figure 3.4: Sign language for word “Ghum”. Head movement is also necessary along with hand gestures.

3.2 Data Preprocessing

We processed the dataset for image labeling. Image labeling is useful when automating the process of generating metadata or making recommendations to users based on details in their images. It helps in object detection. In our case, as we have to detect our hands and so we used image labeling to label the hand’s sign with the particular word that the signs represent. We also applied data argumentation on our dataset.

3.2.1 Data Augmentation

For training models, we employ data augmentation approaches to increase the variety of the data available without having to collect any new data. In this case, we may employ rotation, downsampling, upsampling, padding, flipping, and so on.

Besides, adding noise to a limited dataset can expand it, prevent overfitting, and have a regularizing impact. When a neural network attempts to learn very high-frequency spatiotemporal characteristics or patterns that occur often, the model is generally overfitting. To avoid this problem, data augmentation is carried out utilizing additive Gaussian noise [23].

Data augmentation techniques are used to improve model learning by modifying the current dataset and creating more variants of the pictures. We don't have as many classes to train or test the dataset because we generated our own sign language dataset. So, by augmenting data, we expand our data gathering. Among the techniques frequently used to train neural networks, such as cropping, padding, and horizontal flipping, we utilize downsampling or scaling the picture in this case since we produce the videos in different size frames at first. We also use both hands to make signs. As a result, we use vertical flipping because signs can be produced with either of the two hands, left or right.



Figure 3.5: Sign for the word “Baba” Both the right and left hands are used to sign.

3.2.2 Data Labeling

For image labeling, we used YOLO version 4. We created a folder to put all our selected images for our dataset. After this, we created another folder named “label.” Then we opened our labeling window from the anaconda prompt and opened one frame. By clicking on the “create RectBox ” button, we created our bounding box around our hands in the frame. In the labeling box, we named the signs that we were showing on the frame. We changed the format to YOLO. Then we saved the annotations in our “label” folder. In the folder, two .txt files are going to be created, among which, one is a class file where the names that we gave in labeling will be saved, and another one shows various points of the frame. Those points represent the location, the center point (x and y), the width, and the height of our hands, respectively. We renamed our annotation file the same name as our frame (.jpg file). That is how we labeled each frame of our database one by one.

In our dataset, we have a total of 34 words of sign language in Bangla. For each word, we have 20 frames, and for one word we have created five instances. In total, for one word, we have 100 frames. In our whole dataset, we have 3400 frames for 34 words. Before labeling we manually had to rename all our image files from 1 to

3400 for better results. We kept the image files of the same word side by side while renaming.

We labeled each of them one by one. In the class file, we had the name of the frames that is what sign is shown in the frames, and in the other .txt file, we have all points that represent the height, width, location, center point x, and center point y. The label file name starts from 1 till 3400. The first frame's label file is 1, and the last frame's label file is 3400. As for the location, it starts from 15 and ends at 49. From 15 to 49 we serially label our words. One word consists of 100 frames which means in each location 100 image files exist. The location automatically increases by 1 when we add labels for new words. We used labeling for our labeling and used the default data list that is why our labeling starts from 15 instead of 0 and for 34 words it ends at 49.



Figure 3.6: Labeling for the word “maa”.

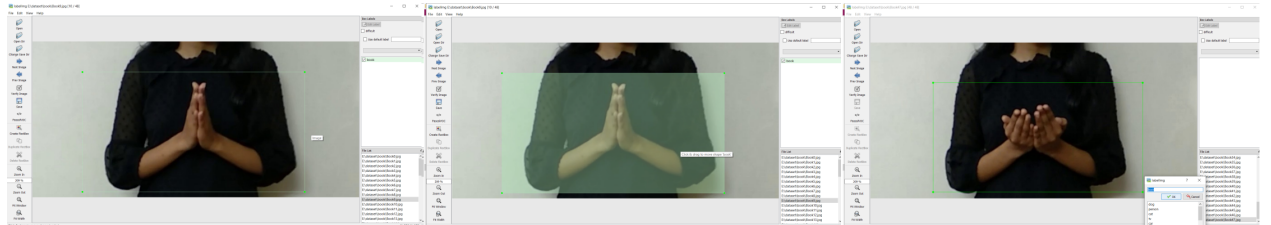


Figure 3.7: Labeling for the word “Boi”.

Chapter 4

Proposed Model

This section is focused on describing our collected dataset for sign language recognition and the suitable algorithms and techniques which are used for training the dataset and classification process are included. The working process is shown in the flowchart below.

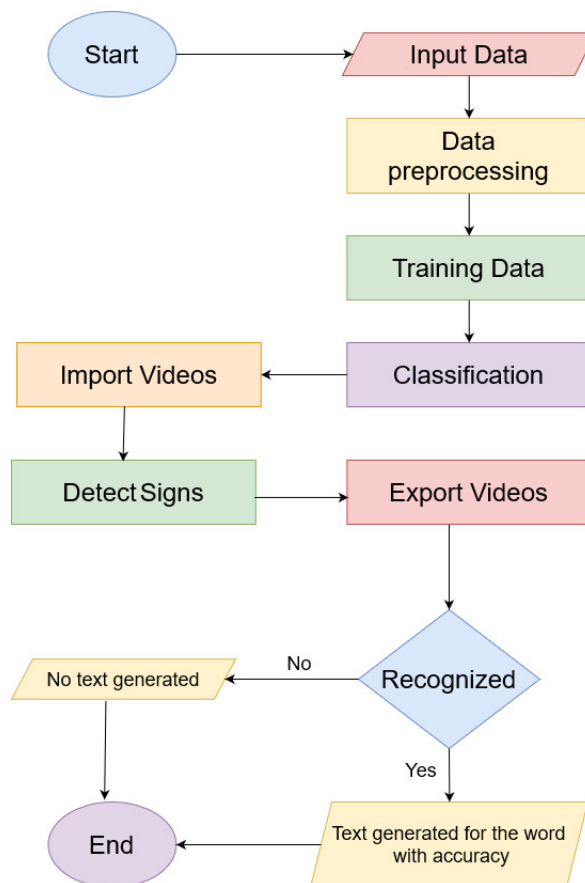


Figure 4.1: Flowchart of the working process.

4.1 Sign Language Detection with Video classification model

Since the dynamic nature of videos, it can be difficult to work with. Videos may be defined as a collection of frames or group of pictures ordered in a specified order. As a result, a video classification problem is similar to an image classification problem. We have to additionally extract frames from the video. In one study the system receives sign language image input which is then processed by the YOLOv3 algorithm. A labeled bounding box is generated when the system will find the trained image in the input [21]. In another study they used YOLOv4 that brings out the features from an input image, produces an object at various scales, and shows bounding-box regions and related classes [17].

Pytorch model of the YOLO network was utilized for hand detection [15]. This framework will also be used for sign detection in our case. We will utilize an object detection technique, the YOLOv5s model from one of the well-known ML framework-PyTorch, in our video classification model.

Transfer learning is utilized in one research with the pretrained YOLOv3 model for Indonesian Sign Language where they used a 4:1 ratio of train set and test set which will evaluate the trained model [20]. We will manually generate the sets. In this case, we will choose YOLOv5s, the smallest and fastest variant accessible.

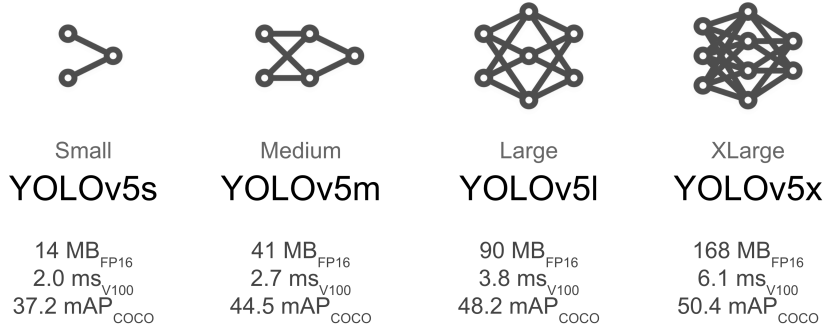


Figure 4.2: Variation in YOLOv5 models
[24]

According to a paper, the feature extraction network of YOLOv3 architectures comprises a Conv Layer, followed by batch normalization, which is important in assisting resolution model process during training, and the Activation function, ReLU [20]. In this work, we will train our model using pre-trained weights of YOLOv5, which will have roughly 30 hyperparameters used for various training settings.

After training, our model will analyze and extract the features from the labeled images. They used the Darknet framework for training and the trained weights were saved there [21]. Pre-trained YOLO from Pytorch weights were used and the data

was transferred to folders, and the annotations were saved [15]. In this section, we will load the weights from the training phase, make predictions, and so evaluate the model.

At testing, a comparison of detected findings and ground truth is done, then calculating the accuracy. This test employs three alternative Intersections over Union (IoU) threshold settings [20]. Similar to these we will produce predictions for our test videos by extracting features from frames and then predicting tags using the pre-trained model.

Following classification, we will have the predicted result to evaluate the model. The evaluation metric, mAP will assess the detection task [20]. In testing, each frame of video data will be detected. When a class is being found in a specific order, it is termed a recognized sign. They developed four metrics to evaluate the performance: Accuracy, Precision, F1-Score and Recall where each detection class was measured independently [17]. We will also follow this approach to determine the class to which the image belongs.

The algorithm just displays the bounding box [15]. However, the sign detection algorithm displays the name of the sign's class together with the bounding box [20], [21]. We shall see a bounding box with the resulting sign in text when the sign has been detected. This bounding box will also demonstrate the sign's accuracy. When a valid sign is given, the screen will display the gesture that is the closest match.

Chapter 5

Implementation and Results

5.1 Sign Language Detection with Video classification model

For our video classification model, we first installed all the dependencies and set up our PyTorch and GPU. Here, we extracted frames from the videos, then utilized feature extractors to take out features from pictures, and then categorized the image according to the recovered features. This model will recognize signs in an iterative process that includes gathering and organizing frames, classifying signs, training our model, and deploying it to make predictions.



Figure 5.1: Training process.

5.1.1 Training Dataset

To begin training, among our 3400 images, we manually generate 80% for the training (2720 images) and 20% for the validation (680 images) sets and organize the directories accordingly. Here our training set has 4 instances whereas the testing set has 1 instance.

Train	4 Instance	2720 Frames	80%
Test	1 Instance	680 Frames	20%
Total	5 Instance	3400 Frames	100%

Table 5.1: Dataset Distribution

Our SL dataset will then be used to train this pre-trained model, YOLOv5. The

batch size is set to 20, the image size is set to 640 pixels, and the pretrained weights ‘yolov5s.pt’ are downloaded automatically from the most recent YOLOv5. We began training with 10 epochs and increased the number progressively. The value remained steady after 50 epochs. As a result, we trained our model for 50 epochs which took 0.617 hours to complete. We employ the Stochastic Gradient Descent (SGD) optimizer with parameter groups, and momentum as one of the hyperparameters. For training, we utilize Mosaic Dataloader, which merges four pictures into one mosaic where we can see train and test jpgs, labels, predictions, and augmentation effects.

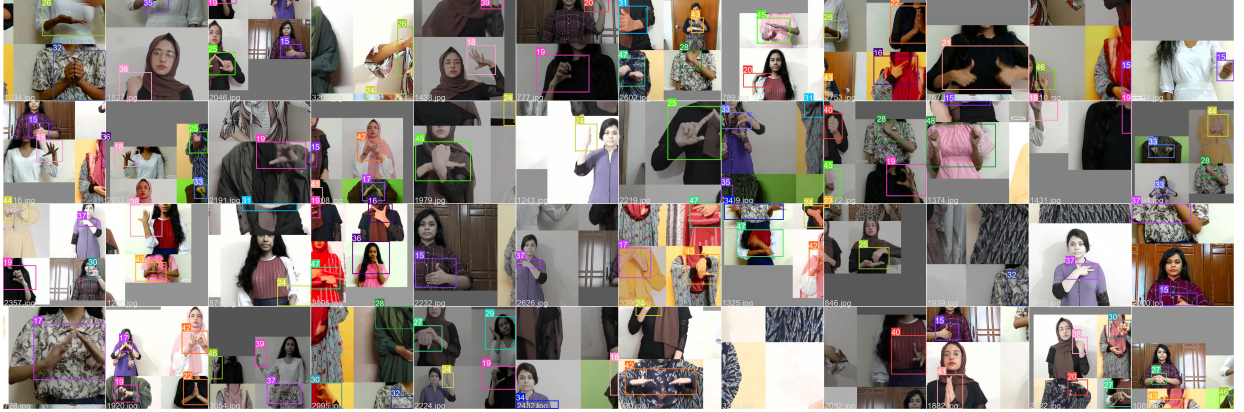


Figure 5.2: Mosaics and labels of train batch 0,1 and 2

Following training, we receive a custom 34-class trained YOLOv5s model with PyTorch Hub in the runs/train/exp directory that contains all of the weights that we will utilize to generate predictions.

Overriding model.yaml nc=80 with nc=49

	from	n	params	module	arguments
0	-1	1	3520	models.common.Conv	[3, 32, 6, 2, 2]
1	-1	1	18560	models.common.Conv	[32, 64, 3, 2]
2	-1	1	18816	models.common.C3	[64, 64, 1]
3	-1	1	73984	models.common.Conv	[64, 128, 3, 2]
4	-1	2	115712	models.common.C3	[128, 128, 2]
5	-1	1	295424	models.common.Conv	[128, 256, 3, 2]
6	-1	3	625152	models.common.C3	[256, 256, 3]
7	-1	1	1180672	models.common.Conv	[256, 512, 3, 2]
8	-1	1	1182720	models.common.C3	[512, 512, 1]
9	-1	1	656896	models.common.SPPF	[512, 512, 5]
10	-1	1	131584	models.common.Conv	[512, 256, 1, 1]
11	-1	1	0	torch.nn.modules.upsampling.Upsample	[None, 2, 'nearest']
12	[-1, 6]	1	0	models.common.Concat	[1]
13	-1	1	361984	models.common.C3	[512, 256, 1, False]
14	-1	1	33024	models.common.Conv	[256, 128, 1, 1]
15	-1	1	0	torch.nn.modules.upsampling.Upsample	[None, 2, 'nearest']
16	[-1, 4]	1	0	models.common.Concat	[1]
17	-1	1	90880	models.common.C3	[256, 128, 1, False]
18	-1	1	147712	models.common.Conv	[128, 128, 3, 2]
19	[-1, 14]	1	0	models.common.Concat	[1]
20	-1	1	296448	models.common.C3	[256, 256, 1, False]
21	-1	1	590336	models.common.Conv	[256, 256, 3, 2]
22	[-1, 10]	1	0	models.common.Concat	[1]
23	-1	1	1182720	models.common.C3	[512, 512, 1, False]
24	[17, 20, 23]	1	145638	models.yolo.Detect	[49, [[10, 13, 16, 30, 33, 23], [30, 61, 62, 45, 59, 119], [116, 90, 156, 196

Model Summary: 270 layers, 7151782 parameters, 7151782 gradients, 16.3 GFLOPs

Figure 5.3: Model summary of YOLO.

As we take a video frame-wise, each word has a varied level of precision. We can observe this in the below figure 15. Some of result of the different training images can be seen in Figure 5.4 and 5.5.



Figure 5.4: The detection of the word “Bhai” with different accuracy.



Figure 5.5: The detection of the words “Shahajjo”, “Bhalo”, “Shikhhok”.

5.1.2 Classification

In our “custom.yaml” file, we specified the number of classes as well as their names. In this context, our YOLOv5 model will automatically locate the labels for each image and learn the classes. Below is the figure 5.6 for predicting in training phase:

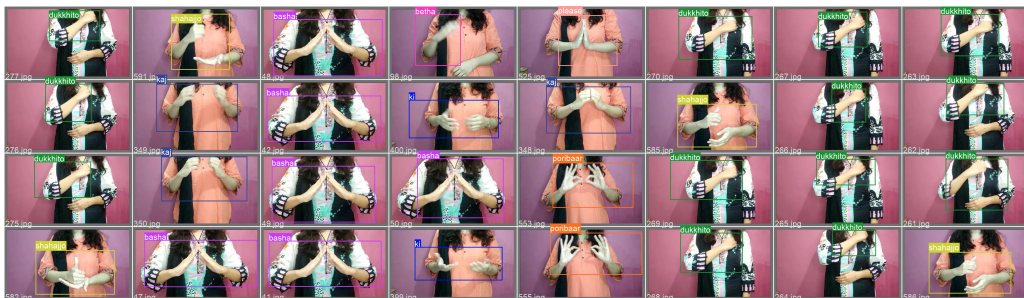


Figure 5.6: Prediction result of classes.

This phase is evaluated using Mean Average Precision (mAP). After this we achieve the following results for each class shown in figure 5.7:

Model Summary: 213 layers, 7142278 parameters, 0 gradients, 16.2 GFLOPs

Class	Images	Labels	P	R	mAP@.5	mAP@.5:.95	100% 17/17	[00:05<00:00, 2.95it/s]
all	680	680	0.192	0.134	0.141	0.0495		
asha	680	20	0.164	0.05	0.0609	0.0129		
baba	680	20	0	0	0	0		
basha	680	20	1	0.444	0.818	0.339		
bati	680	20	0	0	0	0		
betha	680	20	0	0	0	0		
bhai	680	20	1	0	0.023	0.0092		
bhalo	680	20	0.106	0.2	0.15	0.0613		
boi	680	20	0	0	0.00611	0.00242		
bondhu	680	20	0.157	0.1	0.0692	0.0306		
bortoman	680	20	0	0	0	0		
classroom	680	20	0.494	0.293	0.274	0.0667		
cricket	680	20	0.335	0.9	0.737	0.201		
doctor	680	20	0.817	0.65	0.742	0.337		
dukkhito	680	20	0.209	0.1	0.083	0.0232		
ghum	680	20	0.0755	0.2	0.103	0.013		
gorom	680	20	0	0	0	0		
jaoa	680	20	0.0801	0.0481	0.00859	0.00266		
kaj	680	20	0.393	0.1	0.366	0.155		
kharap	680	20	0.897	0.436	0.534	0.164		
ki	680	20	0	0	0.125	0.0197		
kolom	680	20	0.414	0.3	0.319	0.0979		
maa	680	20	0	0	0	0		
naam	680	20	0	0	0.0145	0.00341		
otit	680	20	0	0	0.0156	0.00251		
pakha	680	20	0	0	0	0		
pesha	680	20	0	0	0.00419	0.000419		
please	680	20	0	0	0	0		
poribaar	680	20	0.171	0.6	0.203	0.0943		
raat	680	20	0	0	0	0		
shahajjo	680	20	0.0271	0.05	0.0366	0.0108		
shokal	680	20	0	0	0.013	0.00302		
shundor	680	20	0	0	0.00778	0.000778		
shikkhok	680	20	0.173	0.0946	0.07	0.0297		
thanda	680	20	0	0	0.00457	0.000914		

Figure 5.7: Word Classification after model training.

5.1.3 Sign Detection and Text Generation

After the classification is complete, detection will be performed on each frame of video data during video testing. Our model will match each frame with our dataset.

These detection findings will be compared to the ground truth, and a Confusion Matrix will be created to determine accuracy. We will set the confidence threshold to 0.25.

As we take the videos frame by frame, a bounding box with the name of the sign and the accuracy of recognizing that sign will be displayed in each frame of the testing video. As a result, each word has a varied level of precision. We can observe this in the below figure 5.8:



Figure 5.8: Different accuracy of the word “Basha”.

Some of result of our testing images can be seen in Figure 5.9:

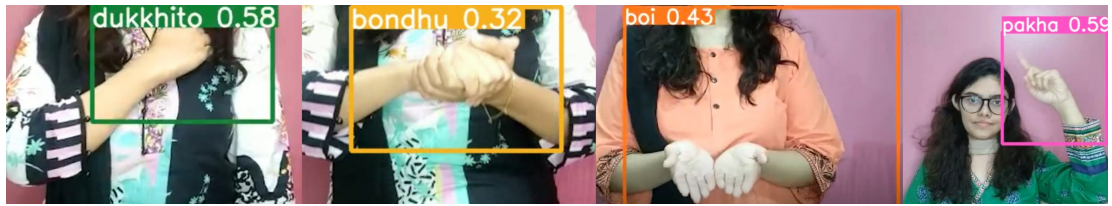


Figure 5.9: Sign detection of the words Dukkhitto, Bondhu, Boi, Pakha

We also put our model through its tests using a different video source. The outcome of the accuracy test is provided underneath in figure 5.10:

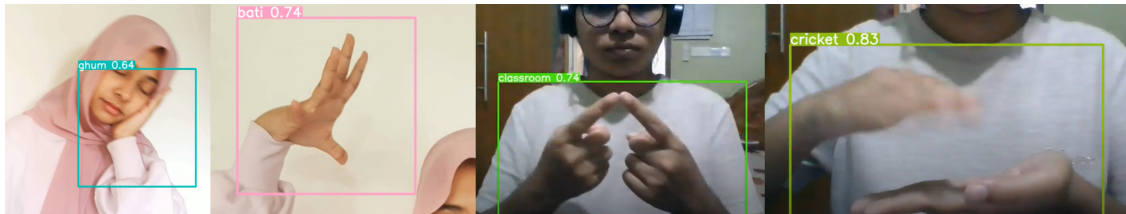


Figure 5.10: Sign detection of the words ghum, bati, classroom, cricket.

Chapter 6

Analysis on Results

6.1 Accuracy

Our approach recognizes signs in videos in one frame that can be identified in one second. Our training model's overall accuracy is 76.29 %, while its testing accuracy is 51.44 %. Object detection tasks are evaluated using Mean Average Precision (mAP). The object classification task will be evaluated using the AP(Average Precision) and AR(Average Recall) evaluation metrics. These are used to calculate the accuracy score and loss score. We can observe this for both train and test sets from the given charts below in figure 6.1:

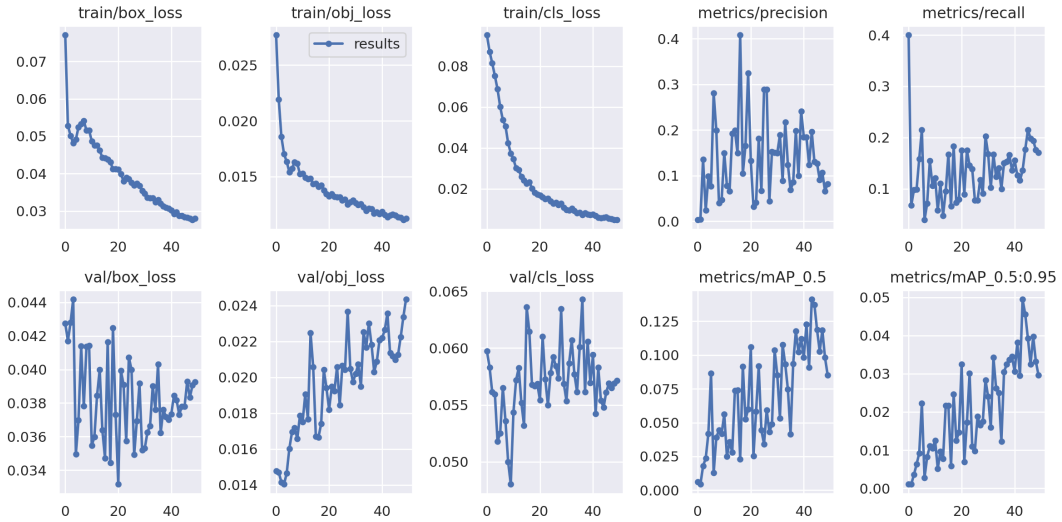


Figure 6.1: Accuracy, Precision, Recall, mAP and loss score charts for train and test sets.

In training, we achieve varying accuracy for 34 words. We take the highest accuracies as well as the average of these accuracies in our calculation. The outcomes of the experiment for all the words are detailed in table 3 below with 4 instances for training:

0 = no value of accuracy, data lost.

No.	Name of Words	1st Instance Highest Accuracy (%)	2nd Instance Highest Accuracy (%)	3rd Instance Highest Accuracy (%)	4th Instance Highest Accuracy (%)	Highest Accuracy Overall (%)	Average Accuracy (%)
1	asha	81	77	82	68	82	77
2	baba	71	76	76	71	76	73
3	basha	82	84	86	87	87	84
4	bati	77	79	86	75	86	79
5	betha	76	86	0	83	86	61
6	bhai	79	80	84	83	84	81
7	bhalo	85	86	85	84	86	85
8	boi	78	75	87	74	87	78
9	bondhu	80	81	83	82	83	81
10	bortoman	64	79	79	83	83	76
11	classroom	80	86	0	89	89	64
12	cricket	73	81	87	91	91	83
13	doctor	74	86	89	81	89	82
14	dukkhito	65	73	74	80	80	73
15	ghum	63	80	80	76	80	74
16	gorom	79	85	83	66	85	78
17	jaoa	67	80	78	70	80	73
18	kaj	71	81	84	85	85	80
19	kharap	77	87	86	89	89	84
20	ki	81	88	87	86	88	85
21	kolom	63	81	82	73	82	74
22	maa	51	71	87	67	87	69
23	naam	68	79	76	75	79	74
24	otit	74	81	75	80	81	77
25	pakha	75	77	69	81	81	75
26	pesha	72	82	92	78	92	81
27	please	68	78	86	81	86	78
28	poribar	77	86	0	89	89	63
29	raat	72	77	80	77	80	76
30	shahajjo	72	85	78	61	85	74
31	shokal	71	73	82	79	82	76
32	shundor	63	81	82	75	82	75
33	shikkhok	59	84	85	85	85	78
34	thanda	67	73	80	71	80	73

Table 6.1: Train Accuracy Table (4 Instance)

The bar chart representing the above training data is shown below in figure 6.2:

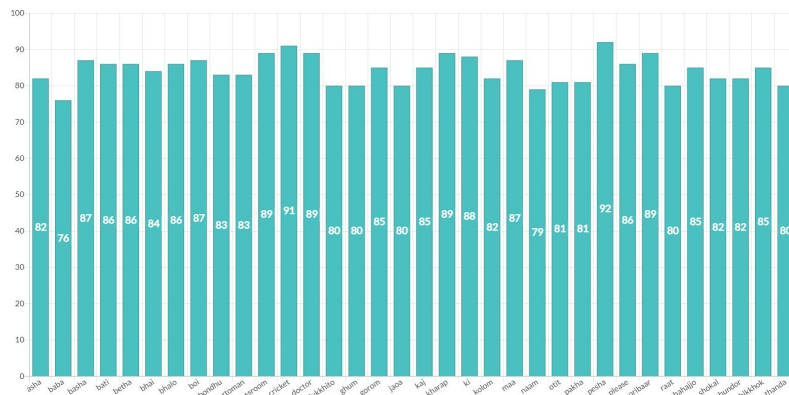


Figure 6.2: Train Accuracy Bar Chart

The highest accuracy for all of the words are shown as graph in figure 24 below for each of the four training instances. When we compare these four, we can see that instance 2 is the best for training our model.

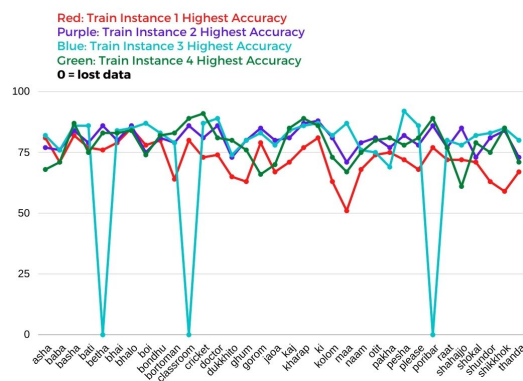


Figure 6.3: Comparison of highest accuracy of all 4 train instances.

In testing, we achieve varying accuracy for 34 words. We take the highest accuracies as well as the average of these accuracies in our calculation. The outcomes of the experiment for all the words are detailed in table 6.2 below 1 instance for testing:

No.	Name of Words	Highest Accuracy (%)	Average Accuracy (%)
1	asha	67	61
2	baba	66	62
3	basha	83	76
4	bati	72	69
5	betha	67	60
6	bhai	33	25
7	bhalo	74	62
8	boi	76	69
9	bondhu	77	72
10	bortoman	58	53
11	classroom	0	0
12	cricket	75	66
13	doctor	63	57
14	dukkhito	63	56
15	ghum	62	57
16	gorom	0	0
17	jaoa	68	64
18	kaj	77	69
19	kharap	79	72
20	ki	0	0
21	kolom	63	56
22	maa	58	53
23	naam	73	68
24	otit	67	59
25	pakha	58	54
26	pesha	0	0
27	please	71	58
28	poribar	63	59
29	raat	67	63
30	shahajjo	54	43
31	shokal	58	53
32	shundor	75	66
33	shikkhok	0	0
34	thanda	72	67

Table 6.2: Test Accuracy Table (1 Instance)

The bar chart representing the above testing data, is shown in figure 6.4:

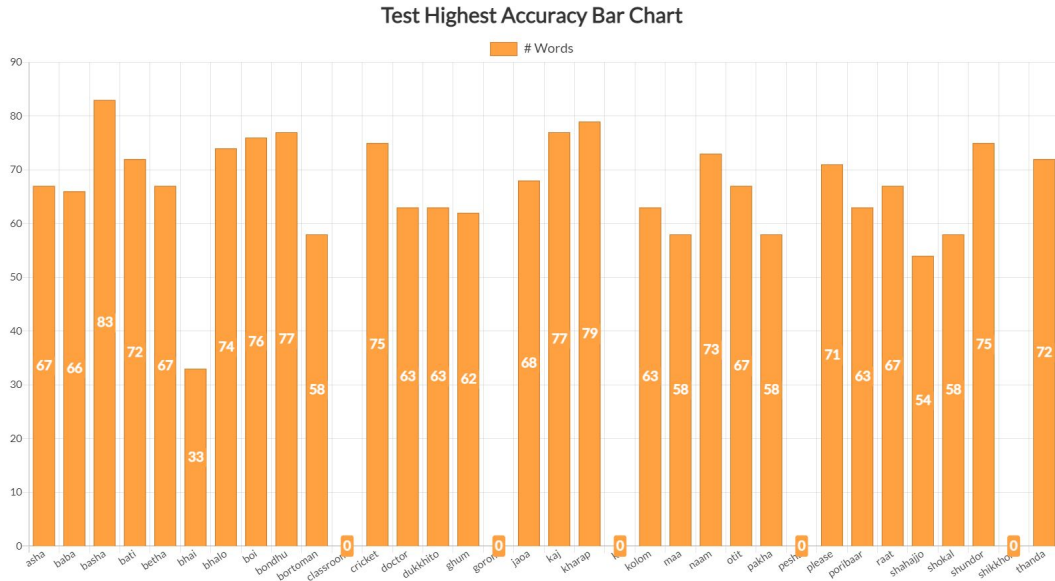


Figure 6.4: Test Highest Accuracy Bar Chart.

6.2 Analysis of training and testing sets

The graph below in figure 6.5 is drawn from the upper tables of train and test average accuracy. We may deduce from the graph that for all the 34 words our average training accuracy is higher than our average testing accuracy. We can see below that several of the words were not caught throughout our testing process. Such as “classroom”, “gorom”, “ki”, “pesha”, “shikkhok”.

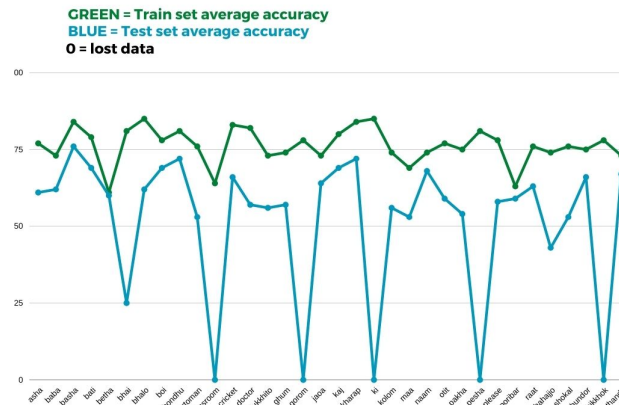


Figure 6.5: Comparison between train-test average accuracy.

All of the detections provided the correct class or the classes that our model could not identify, as shown in Table 6.3. In both the training and testing phases, we determined the percentage of classes our model was able to recognize and the percentage of classes our model failed to detect.

	Train Data	Test Data
Total words	34	34
Total Instance	4	1
Total video	136	34
Accurately Recog- nized	133	29
Recognition Percent- age	97.79%	85.29%
Loss	3	5
Loss Percentage	2.21%	14.71%

Table 6.3: Percentage of accuracy and loss in train and test data

6.3 Evaluation

We worked on a self-made dataset of 34 words. Which is divided into train and test for our whole process of work. We worked on a video classification model that is trained on YOLO and PyTorch and would detect from direct videos that we import in our model. We get our results that 97.79 % of our training set videos can be properly recognized and only 2.21 % gets lost. In the test data, 85.29 % can be recognized and 14.71 % gets lost.

Causes of data loss:

- If the videos that we import in our model are not in format .mp4, .mov, .avi, .mkv, .wmv, then the content of the video will not be recognized.
- If the videos that import are vertically aligned then the sign shown in the video might also not get recognized.

Causes of low accuracy:

- Low accuracy might occur in case of low video quality.
- If the sign is shown too far from camera or too close into camera then the accuracy might be low.
- In our self made dataset, there might be a few mistakes in labeling for which low accuracy can occur.



Figure 6.6: vertically aligned video.



Figure 6.7: Video taken through webcam, too close to camera, cannot detect the sign for “shahajjo” and “doctor”.

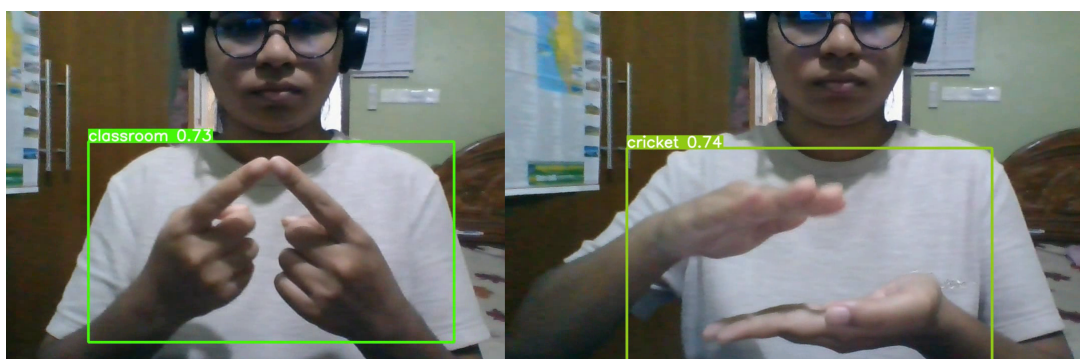


Figure 6.8: Video taken through webcam, full sign can be seen and detected perfectly.

Chapter 7

Conclusion and Future Work

7.1 Conclusion

In the end, we would like to mention that all the factual assessments given by us are based on the paper we read correlates with the Bangla sign to text generation system are all different types of neural networks. The reviewed literature and working flow, we wrote, are completely based on the paper we have studied connected to our field to understand how we should approach and get a clear view of those fields which we will use for our further research. Not only that, but we have also mentioned the dataset which we used to generate a clear idea about those fields and how it works and that's how we implemented it in our research. Basically, we built our own model where we made our own dataset for training and classifying. We have completed all the processes like taking input of the Data, labeling the data, augmentation, training and classification of the data etc. Finally, we'd like to point out that the techniques and models we discuss in this study are based on a publication we read for our research. It is a preliminary idea and needs to be more developed to get the highest accuracy output.

7.2 Future Work

In the future, we will endeavor to improve the efficiency of our system. As previously indicated, we constructed our own dataset for word recognition utilizing 34 words. We will increase the amount of words in our dataset in the future. We aim to use it for sentence recognition. In other languages, such as American Sign Language, sentences can be recognized in real time, however in the case of Bangla, not much work has been done on sentence recognition in real time. As a consequence, for future work, our main motive will be to work on sentence recognition in real time with a larger dataset. Nonetheless, our main objective always will be to facilitate the persons who have speech and hearing problems.

Bibliography

- [1] *Bangla sign language dictionary [6lkvezd3e204]*, Mar. 1997. [Online]. Available: <https://doku.pub/documents/bangla-sign-language-dictionary-6lkvezd3e204>.
- [2] F. Z. Siddiqua, *The silent conversation*, Mar. 2015. [Online]. Available: <https://www.thedailystar.net/the-silent-conversation-37929>.
- [3] F. N. Arko, N. Tabassum, T. R. Trisha, and F. Ahmed, Apr. 2017. [Online]. Available: <http://dSPACE.bracu.ac.bd/xmlui/handle/10361/8216>.
- [4] G. A. Rao, P. V. V. Kishore, D. A. Kumar, and A. S. C. Sastry, “Neural network classifier for continuous sign language recognition with selfie video,” *Far East Journal of Electronics and Communications*, vol. 17, pp. 49–71, 2017.
- [5] C. Andrew and R. Fiona, “A survey on video classification using action recognition,” *International Journal of Engineering and Technology(UAE)*, vol. 7, pp. 89–93, May 2018. DOI: 10.14419/ijet.v7i2.31.13404.
- [6] J. Huang, W. Zhou, Q. Zhang, H. Li, and W. Li, *Video-based sign language recognition without temporal segmentation*, 2018. arXiv: 1801.10111 [cs.CV].
- [7] A. P. Jana, A. Biswas, and Mohana, “Yolo based detection and classification of objects in video records,” in *2018 3rd IEEE International Conference on Recent Trends in Electronics, Information Communication Technology (RTE-ICT)*, 2018, pp. 2448–2452. DOI: 10.1109/RTEICT42901.2018.9012375.
- [8] S. Masood, A. Srivastava, H. C. Thuwal, and M. Ahmad, “Real-time sign language gesture (word) recognition from video sequences using cnn and rnn,” in *Intelligent Engineering Informatics*, Springer, 2018, pp. 623–632.
- [9] S. Ahmed, M. R. Islam, J. Hassan, *et al.*, *Hand sign to bangla speech: A deep learning in vision based system for recognizing hand sign digits and generating bangla speech*, 2019. arXiv: 1901.05613 [cs.CV].
- [10] H. M. Mariappan and V. V. Gomathi, “Real-time recognition of indian sign language,” *2019 International Conference on Computational Intelligence in Data Science (ICCIDS)*, pp. 1–6, 2019.
- [11] A. Paszke, S. Gross, F. Massa, *et al.*, *Pytorch: An imperative style, high-performance deep learning library*, 2019. arXiv: 1912.01703 [cs.LG].
- [12] M. A. Rahaman, M. Jasim, M. Ali, and M. Hasanuzzaman, “Bangla language modeling algorithm for automatic recognition of hand-sign-spelled bangla sign language,” *Frontiers of Computer Science*, vol. 14, 2019.

- [13] P. Roy, S. M. Uddin, M. A. Rahman, M. M. Rahman, M. S. Alam, and M. S. Rashid Mahin, "Bangla sign language conversation interpreter using image processing," in *2019 1st International Conference on Advances in Science, Engineering and Robotics Technology (ICASERT)*, 2019, pp. 1–5. DOI: 10.1109/ICASERT.2019.8934614.
- [14] T. M. Tan, A. M. Mondol, N. Nawal, and S. Ahmed, Aug. 2019. [Online]. Available: <http://dspace.bracu.ac.bd/xmlui/handle/10361/13638>.
- [15] A. Ananthanarayan, "Hand detection and body language recognition using yolo," Ph.D. dissertation, 2020.
- [16] D. Li, C. R. Opazo, X. Yu, and H. Li, *Word-level deep sign language recognition from video: A new large-scale dataset and methods comparison*, 2020. arXiv: 1910.11006 [cs.CV].
- [17] D. Talukder and F. Jahara, "Real-time bangla sign language detection with sentence and speech generation," in *2020 23rd International Conference on Computer and Information Technology (ICCIT)*, 2020, pp. 1–6. DOI: 10.1109/ICCIT51783.2020.9392693.
- [18] A. Tunga, S. V. Nuthalapati, and J. Wachs, *Pose-based sign language recognition using gcn and bert*, 2020. arXiv: 2012.00781 [cs.CV].
- [19] K. Yin, "Sign language translation with transformers," *ArXiv*, vol. abs/2004.00588, 2020.
- [20] S. Daniels, N. Suciati, and C. Fathichah, "Indonesian sign language recognition using yolo method," *IOP Conference Series: Materials Science and Engineering*, vol. 1077, p. 012 029, Feb. 2021. DOI: 10.1088/1757-899X/1077/1/012029.
- [21] S. Ibrahim, I. Mohtar, and Z. Ahmad, "A real time malaysian sign language detection algorithm based on yolov3," *International Journal of Recent Technology and Engineering*, vol. 8, p. 651, Apr. 2021. DOI: 10.35940/ijrte.B1102.0982S1119.
- [22] B. M, J. J, K. M, and S. S, "Sign language translator using yolo algorithm," in Nov. 2021, ISBN: 9781643682181. DOI: 10.3233/APC210136.
- [23] M. Rehman, F. Ahmed, M. Khan, *et al.*, "Dynamic hand gesture recognition using 3d-cnn and lstm networks," *Computers, Materials and Continua*, Sep. 2021. DOI: 10.32604/cmc.2021.019586.
- [24] *Multi-gpu training*. [Online]. Available: <https://docs.ultralytics.com/tutorials/multi-gpu-training/>.