Bridging the Communication Gap: An AI-Driven word-level Bangla Sign Language Interpreter

By
MD Saidur Rahaman
213-15-4489

FINAL YEAR DESIGN PROJECT REPORT

This Report Presented in Partial Fulfillment of the Requirements for the **Degree of Bachelor of Science in Computer Science and Engineering**

Supervised by

Md. Firoz Hasan
Senior Lecturer
Department of Computer Science and Engineering Daffodil International University

Co-Supervised by

Md Umaid Hasan
Senior Lecturer
Department of Computer Science and Engineering Daffodil International University

DAFFODIL INTERNATIONAL UNIVERSITY
Dhaka, Bangladesh

September 17, 2025

APPROVAL

This Project titled **"Bridging the Communication Gap: An AI-Driven word-level Bangla Sign Language Interpreter"**, submitted by **MD Saidur Rahaman**, ID No: **213-15-4489** to the Department of Computer Science and Engineering, Daffodil International University has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of B.Sc. in Computer Science and Engineering and approved as to its style and contents. The presentation has been held on **17 September, 2025**.

BOARD OF EXAMINERS

Dr. Bimal Chandra Das (BCD) — **Chairman**
Professor and Dean (in-charge)
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

Mr. Shah Md Tanvir Siddiquee (SMTS) — **Internal Examiner**
Assistant Professor
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

Md. Hasanuzzaman Dipu (MHD) — **Internal Examiner**
Assistant Professor
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

Nazibur Rahman — **External Examiner**
Technical Lead - Database Administrator
Telenor - Grameen Phone Account

©Daffodil International University

DECLARATION

We hereby declare that this project has been done by us under the supervision of **Md. Firoz Hasan, Senior Lecturer**, Department of Computer Science and Engineering, Daffodil International University. We also declare that neither this project nor any part of this project has been submitted elsewhere for the award of any degree or diploma.

Supervised by:

15.9.25

Md. Firoz Hasan
Senior Lecturer
Department of Computer Science and
Engineering Daffodil International University

Co-Supervised by:

15.09.2025

Md Umaid Hasan
Senior Lecturer
Department of Computer Science and
Engineering Daffodil International University

Submitted by:

Saidur 17.09.2025

MD Saidur Rahaman
Student ID:213-15-4489
Department of Computer Science and
Engineering Daffodil International University

ACKNOWLEDGEMENTS

This work would not have been possible without the support and contributions of many individuals over the past two semesters. We are deeply grateful to everyone who has assisted us in one way or another.

First, we express our heartfelt thanks and gratefulness to the almighty for His divine blessing making it possible for us to complete the **Final Year Design Project(FYDP)** successfully.

We are grateful and wish our profound indebtedness to **Md, Firoz Hasan**, **Senior Lecturer**, Department of Computer Science and Engineering, Daffodil International University, Dhaka, Bangladesh. Deep knowledge and keen interest of our supervisor in the field of **Machine Learning** to carry out this project. His endless patience, scholarly guidance, continual encouragement, constant and energetic supervision, constructive criticism, valuable advice, reading many inferior drafts, and correcting them at all stages have made it possible to complete this project.

We would like to express our heartfelt gratitude to the Head of the Department of Computer Science and Engineering, for his kind help in finishing our project and also to other faculty members and the staff of the Department of Computer Science and Engineering, Daffodil International University.

We would like to thank our entire course-mates at Daffodil International University, who took part in this discussion while completing the coursework.

Finally, we must acknowledge with due respect the constant support and patience of our parents.

ABSTRACT

The present paper is a research project report creating a real-time Bangla Sign Language (BdSL) recognition system in a two-phase approach. The initial phase concentrated on developing a state-of-the-art baseline with our proposed hybrid Conv1D-BiLSTM architecture including the Attention mechanism with an accuracy of 97.3% on the published 60-word BDSLw60 v2 dataset, which is higher than the previously published benchmarks. The second stage of research was devoted to the important issue of vocabulary development, and a new set of signs of 39 signs, which had been collected by hand, was presented. In order to address the dramatic accuracy drop experienced with first fine-tuning, we designed and adopted a professional quality multi-stage fine-tuning policy that includes class weights and gradual unfreezing. This sophisticated procedure was very practical and the final and strong model reached 93 percent total accuracy in the whole vocabulary (99-word). This last model is implemented on a real-time Flask web application, which translates signs directly to text, showing an end-to-end cycle between a validation of the baseline to a scalable and real-world application.

Keywords--Bangla Sign Language, Deep Learning, State-of-the-art, transfer learning, real-time system, LSTM, attention mechanism, computer vision.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

This chapter provides a comprehensive introduction to the project, establishing the background and problem statement for the research. It will cover the motivation for this work, a clear enumeration of the project objectives, a summary of the methodology, the expected outcomes, and a guide to the organization of this report.

1.1 Introduction

Communication is a key element of human interaction, but people with hearing impairments (the deaf and the hard-of-hearing) face major barriers. In Bangladesh, where Bangla Sign Language (BdSL) is the language most commonly used by this population group, insufficient fluency among the general population presents a hindrance in education, government services, and life overall. Although Artificial Intelligence, specifically deep learning has demonstrated tremendous potential in the creation of automated sign language interpreters, one of the largest issues with the concept is that most scholarly models are trained on small and fixed vocabularies. Another important challenge is the ability to successfully and efficiently increase the vocabulary of an already trained model without the need to undergo a full and computationally intensive retraining process. This problem is directly tackled in this project since it will generate a high-accuracy baseline model and subsequently study a way to scale up its vocabulary.

1.2 Motivation

This project was mostly driven by the need to solve the famous deep learning problem of catastrophic forgetting and the negative performance transfer in the context of transfer learning. It is common that a highly optimized model will forget the original classes it has learned on when it is fine-tuned with new classes. We were driven to study and apply a fine-tuning strategy that is up-to-date, in the context of a sign language recognition task, and capable of accommodating new vocabulary without disrupting the already acquired knowledge. Solving this problem provides a proven and scalable framework, benefiting the 53development of more practical and extensive assistive technologies that can grow and adapt over time.

1.3 Objectives

The particular aims of this research project were as follows:

1. To create a baseline deep learning model based on an Attention-based hybrid Conv1D-BiLSTM architecture and achieve the state-of-the-art accuracy on the public

BDSLw60 v2 dataset, making it even more accurate than current benchmark.

2. To gather, process and curate a fresh, custom data collection of 39 new Bangla signs to enable study into the growth of vocabulary.

3. To study, apply and test a high-quality fine-tuning procedure, including methods such as class weighting and gradual unfreezing, to successfully introduce the new vocabulary to the pre-trained model.

4. To hit high overall accuracy with the final, expanded 99-word vocabulary, to prove that the fine-tuning strategy was successful.

5. To implement the final high-precision model into an efficient, real-time web application that can work with changing video inputs as a proof-of-concept.

1.4 Methodology

The implementation of the project involved taking a two-phase approach. The baseline model was trained in the 60-word dataset BDSLw60 v2 in the first phase. The video data was run through Google MediaPipe to detect 1,662 keypoints per frame, which were inputted into a hybrid Conv1D-BiLSTM with Attention network, resulting in 97.3 percent accuracy. The second stage was concerned with vocabulary growth. The 39 newly collected words were extracted to features. The baseline model was then fine-tuned in a multi-stage process with the help of class weights to manage data imbalance and a gradual unfreezing approach with a well-chosen learning rate. This enabled the model to learn the new classes without forgetting the original ones, and the final model had 99 classes with 93 percent overall accuracy. The implementation of the project involved taking a two-phase approach. The baseline model was trained in the 60-word dataset BDSLw60 v2 in the first phase. The video data was run through MediaPipe to detect 1,662 keypoints per frame, which were inputted into a hybrid Conv1D-BiLSTM with Attention network, resulting in 97.3 percent accuracy. The second stage was concerned with vocabulary growth. Features were extracted from the 39 newly collected words. The baseline model was then fine-tuned in a multi-stage process with the help of class weights to manage data imbalance and a gradual unfreezing approach with a well-chosen learning rate. This enabled the model to learn the new classes without forgetting the original ones, and the final model had 99 classes with 93 percent overall accuracy.

1.5 Project Outcome

The potential outcomes of this work are:

1. An advanced baseline model of the BDSLw60 dataset, which achieved a reported accuracy of 97.3 and could be used as a reference point in future studies.

2. An empirical and established system of achieving success in extending the range of vocabulary of a sign language recognition model, which may be used in other similar undertakings.

3. Strong, deep learning model that effectively identifies an extended 99-word Bangla sign vocabulary.

4. A real-time web application which also functions as a workable demonstration of concept and an actual communication tool.

1.6 Organization of the Report

In this report, there are six chapters. In chapter 1, an introduction to the project is presented, which includes the problem statement, motivation, objectives, and a summary of the work. Chapter 2 examines related literature with a view to identifying the major benchmarks of the BDSLw60 dataset. Chapter 3 outlines the methodology of the paper in more detail, including preparations of the dataset, feature extraction, the structure of the model, and the advanced fine-tuning strategy. Chapter 4 describes the experimental findings and gives an in-depth analysis of the base as well as the performance of the final fine-tuned model. Chapter 5 explains how the final model was deployed as a real-time web application. Lastly, Chapter 6 provides a conclusion to the report, summing up the accomplishments, and outlining the shortcomings of the present system, as well as proposing future work directions.

Chapter 2

Background

This chapter gives an in-depth overview of both the academic and technological environment in which this project is operating. It starts with an introduction to the development of the sign language recognition systems, followed by a detailed literature review of the most important researches, analysis of existing applications, and finally, gap analysis, which defines very clearly the new contributions of this paper.

2.1 Introduction

In order to understand the methodology and contributions of this project, it is necessary first to set the technological context in which it operates. In this section, we will present the background knowledge about the different technologies and concepts that are the basis of our sign language recognition system. We will cover the evolution of recognition systems, the motivation behind our feature extraction scheme, the individual roles of the various components in our deep learning hardware, as well as the main principles of the transfer learning problem that we wanted to address.

Early SLR solutions have tended to use sensor-based devices (such as data gloves or motion-capture suits). While some of these systems were able to provide high accuracy by recording accurate kinematic data, they relied on expensive, cumbersome, and impractical hardware that precluded day-to-day, real-world use. Vision-based systems with conventional cameras have become the paradigm of modern SLR, due to the ubiquity of webcams and smartphones. Following the vision-based approach, this project aims at developing such a system in an easily accessible and no-special-equipment-required manner.

Landmark-Based Feature Extraction: A Better Alternative to Raw Video
The high-dimensionality and variability of raw video data is one of the major issues in vision-based SLR. A model trained on the raw pixels is extremely sensitive to irrelevant effects such as background clutter, lighting conditions, the signer's clothes, and camera angle. To address this, our system uses a landmark-based feature extraction implementation powered by Google's MediaPipe.

Instead of analyzing the whole video frame, MediaPipe's holistic solution precisely detects and tracks a 3D Skeletal structure of the signer in real time. This consists of 33 pose landmarks, 21 hand landmarks per hand and 468 facial landmarks. These landmarks are transformed into a normalized coordinate system resulting in a feature vector (in our case, of 1,662 points) which describes the signer's posture and facial expressions.

This technique is a powerful feature engineering and dimensionality reduction tool with a

few core benefits:

1. Robustness: It is much less sensitive to background noise and lighting
2. Efficiency: Rather than processing hundreds of thousands of pixels at inference time, the model processes a small vector of coordinates and thereby training and inference is much faster.
3. Focus: by default, it draws the model's attention to the most relevant information, namely, the human signer.

A sign gesture is a dynamic sequence of motion with a well-defined temporal structure rather than a static image.
We use a hybrid deep learning structure, particularly designed to interpret this spatio-temporal information:

1. 1D Convolutional Neural Network (CNN): The first layer takes the 1,662-point vector of features for each frame. The CNN is a spatial pattern recognizer that learns to identify salient patterns of landmarks, such as the shape of a hand, the inter-hand distance, or a facial expression at an instant in time.

2. Bidirectional Long Short-Term Memory (Bi-LSTM): The sequence of spatial patterns extracted by the CNN is then passed to Bi-LSTM layers. LSTMs are an example of a recurrent neural network that can learn from sequential data by retaining a "memory" of previous data. By adding the ability to rewind the model, bidding it from the end to the beginning, the model acquires a more complete understanding of the gesture trajectory from its beginning to the end, and from the end to the beginning.

3. Attention mechanism: Not all frames are important in a sign. Some are transitional movements and others contain the core, discriminative part of the gesture. The Attention layer is an advanced mechanism which learns to give a weight or "importance score" to each frame in the sequence. This enables the model to dynamically pay attention to the most relevant moments for making its final prediction, enormously increasing its accuracy for complex or subtle signs.

The second stage of our research is based on Transfer Learning, which is a technique that uses a model pre-trained on a source task (the 60 words) and then applies it to a target task (the full 99 words) that is related to the source task. The first advantage is a very large reduction in training time and training data. However, this comes at the cost of the risk of "catastrophic forgetting," whereby the model's performance on the original task is reduced while it learns the new one. Our framework, as explained in the next chapter, specifically aims at reducing this risk via sophisticated fine-tuning techniques.

2.2 Literature Review

Sign Language Recognition (SLR) has undergone a dramatic change in recent years from cumbersome sensor-based approaches to more flexible and vision-based SLR systems based on deep learning. Our research is framed within this contemporary paradigm which is informed by a rich literature from the wider SLR community while considering the particular issues presented by BdSL. In this review, we will take a path from the oldest to the most recent tasks, and we will discuss the main methodological advances which inspired our own approach to architecture and fine-tuning for BSLR to move from the simplest to the most complex word-level recognition task.

Bangla Sign Language Recognition (BSLR) research first gained momentum by addressing the basic task of recognizing static gestures (alphabets and numerals). This initial work laid an important foundation towards proving the feasibility of learning the specific handshapes and features of Bangla signs in computer vision models. An excellent study by Yasir et al. [11] designed a custom convolutional neural network (CNN) that attained an outstanding 98.7 percent accuracy on the 36 alphabets and 10 numerals of BdSL. This was good evidence that CNNs were well suited for learning the spatial details of Bangla signs. Taking the next step, Rakib et al. [12] introduced a CNN architecture that is much more lightweight for the same task, achieving an accuracy of 97.58%, proving that even computationally light models can be highly effective. Further confirmation was provided by Karim et al. [16], in which a deep CNN also achieved near-perfect accuracy on a customised alphabet data set. Together, these studies provided a proof-of-concept for applying CNNs to Bangla signs, but the real challenge is to recognize full, moving words, which are fluid, sequential gestures rather than static images.

The transition from static alphabets to dynamic word-level reading needed more complex models and, importantly, a standard video dataset. The release of the BDSLw60 dataset was a major catalyst for this line of research in Bangladesh. The paper of Jubayer et al. [1] was a seminal study that gave the first overall performance benchmark on this dataset. They went deep, trying out a wide variety of architectures, and achieving a state-of-the-art accuracy of 95.83% with a Bidirectional GRU model. This paper provided the gold standard against which our baseline model would need to perform at and above. Further research on the same dataset by Mondal et al. [2] with a different CNN-based approach obtained a respectable 93.67% accuracy. These papers were key in getting BSLR research from single characters into the much more practical, word-level challenge.

Our own methodology was influenced strongly by general trends in the global SLR community, specifically the trend to adopt landmark-based features. Instead of feeding computationally heavy raw video to a model, modern approaches often rely on pose estimation to generate a lightweight robust skeleton representation of the signer. A very relevant work on this topic is the one by De Coster et al. [7], where they applied state-of-the-art Transformer networks directly on these keypoints for continuous sign

language recognition, proving how powerful the approach is. Furthermore, the use of rich sets of landmarks is a common theme. The landmark work of Koller et al. [5] on German Sign Language was one of the first studies to show quantitatively that by using a combination of features from the face and body with hand information we can gain important linguistic information which strongly enhances the recognition rates. This work directly supports our choice to use MediaPipe's holistic solution (that captures this whole range of expressive features) instead of a hand-only solution. We also chose the hybrid CNN-LSTM architecture as our method was confirmed by the work of Athar et al. [4] who implemented a similar combination for word-level ASL and ensured the robustness of the hybrid CNN-LSTM architecture across different sign language systems.

Finally, the fundamental research contribution of our project (vocabulary expansion) is based on the concepts of fine-tuning and transfer learning. While not specific to SLR, Howard and Ruder's ULMFiT seminal paper [6] was the one which trembled the foundations of the way practitioners solve this problem. They have proposed methods such as "gradual unfreezing," which provides a strong theoretical foundation for transferring pre-trained models to new tasks with reduced "catastrophic forgetting" (the well known problem that the pre-trained model forgets its learned knowledge). Their ideas were a direct inspiration for the design of our successful multi-stage fine-tuning protocol. Finally, Li et al. [8] provide echoes of our own motivation for the custom data collection. Their work on the creation of WLASL, a new large-scale ASL dataset, highlighted a key reality in the field, that progress in deep learning in SLR is intrinsically tied to the availability of large and diverse datasets, which motivated our work to go beyond the available BDSLw60 vocabulary.

Table 2.1: Summary of Literature Reviewed.

Author(s)	Year	Title	Methodology	Data Type	Key Findings / Accuracy	Relevance to this Project
Jubayer et al. [1]	2021	A Deep Dive into Word-level Bangla Sign Language Recognition	CNN, LSTM, B-GRU	BDSLw60 Videos	Established benchmark of **95.83%** on the BDSLw60 dataset.	**Primary Benchmark.** Our baseline model surpassed this SOTA.
Yasir et al. [11]	2021	An Automated System for Recognition of Bangla Sign Language	Custom CNN	BdSL Alphabet/Numeral Images	Achieved 98.7% accuracy on static BdSL alphabet recognition.	Validated the use of CNNs for learning spatial features of Bangla signs.
Rakib et al. [12]	2022	A Lightweight CNN for Bangla Sign Language Alphabets Recognition	Lightweight CNN	BdSL Alphabet Images	Achieved 97.58% with an efficient model.	Showed that even efficient models can be effective for BdSL.
Mondal et al. [2]	2021	Bangla Sign Language Recognition using CNN	CNN	BDSLw60 Videos	Achieved 93.67% accuracy on BDSLw60.	Provided another validation point for the BDSLw60 dataset.
Karim et al. [16]	2022	Recognition of Bangla Sign Language Alphabets using a Deep CNN	Deep CNN	BdSL Alphabet Images	Achieved 99.2% accuracy on a custom BdSL alphabet dataset.	Further confirmed the high performance of CNNs on static BdSL gestures.
Howard & Ruder [6]	2018	Universal Language Model Fine-tuning for Text Classification	Transfer Learning	Text Data	Introduced key **fine-tuning** techniques like gradual unfreezing.	**Core Methodological Inspiration** for our vocabulary expansion phase.
De Coster et al. [7]	2021	Sign Language Recognition with Transformer Networks	Transformers	Pose Keypoints	Demonstrated the power of advanced models on skeletal data for SLR.	Validated the landmark-based feature extraction approach.
Koller et al. [5]	2016	Deep Sign: Hybrid CNN-HMM for Continuous Sign Language Recognition	CNN-HMM	Multi-modal Video	Proved that combining face/body features improves accuracy.	Justified our use of MediaPipe's holistic solution over just hands.
Li et al. [8]	2020	Word-level Deep Sign Language Recognition from Video	3D CNN	New Large-scale ASL Dataset	Highlighted the critical need for larger datasets for generalization.	Motivated our custom data collection to expand the vocabulary.

Author(s)	Year	Title	Methodology	Data Type	Key Findings / Accuracy	Relevance to this Project
Athar et al. [4]	2022	Word-level sign language recognition using a deep CNN-LSTM model	CNN-LSTM Hybrid	ASL Videos	Validated the effectiveness of the CNN-LSTM architecture for word-level SLR.	Supported our choice of a similar hybrid architecture.
Rastgoo et al. [3]	2021	3D-CNN-based short-term sign language recognition	3D CNNs	RGB-D Videos	Showed 3D CNNs are effective for spatio-temporal features.	Represents an alternative, more computationally heavy methodology.
Tasnim et al. [17]	2021	A Real-Time Translator for Bangladeshi Sign Language	CNN	BdSL Alphabet/Number Images	Developed a real-time system for static BdSL gestures.	Showed early interest in deploying real-time BdSL systems.
Al-Hammadi et al. [13]	2020	Deep learning-based approach for Arabic sign language recognition	CNN	Arabic SL Images	Achieved 96.5% on static Arabic sign alphabet recognition.	Shows the applicability of CNNs to other sign languages.
Elakkiya et al. [14]	2023	A survey on various methods for Indian Sign Language recognition	Literature Survey	ISL Datasets	Provided a comprehensive overview of methods and challenges for ISL.	Gives context to SLR research in the subcontinent.
Adaloglou et al. [10]	2020	A Comprehensive Study on Deep Learning-based Methods for SLR	Literature Survey	Various SLR Datasets	Gave a broad overview of state-of-the-art methods across many languages.	Provides a general academic context for the field.

2.2.1 Similar Applications

Whilst there is active academic research into sign language recognition, there is surprisingly few real-time interpreter applications available for public use. The current market is still highly biased towards educational tools, mostly for American Sign Language (ASL) that serves as an indication of the gap we aim to fill with our project. In fact, an analysis of the most representative applications shows a clear difference between tools aimed at the learning of sign language and the ones, such as ours, aimed at real-time interpretation.

Hand Talk: Possibly the most popular application in this space, Hand Talk is a powerful mobile app that converts spoken and written text into sign language. This aspect of the app includes a friendly 3D avatar, Hugo, that performs the signs. While technically impressive, its primary function is text-to-sign, which is essentially the exact opposite of what our system is. It is a tool for the non-signer to communicate with a signer, while our project is a tool for the signer to communicate with a non-signer. It mainly has support for ASL and Brazilian Sign Language (Libras), which again highlights the absence of tools for languages such as BdSL.

Sign.io: This mobile app is one of the very few direct competitors of ours that is based on real-time interpretation. It uses the camera of the phone to do ASL-to-text translation. Its existence proves this demand for such technology. However, it focuses solely on ASL, a language for which a much larger number of data and resources are available than for BdSL. Our project responds to the needs of a different linguistic and geographical community.

The ASL App and the SignSchool: These two platforms are powerhouses in the realm of sign language learning. They act as rich digitized libraries with thousands of high-quality, pre-recorded videos of signs, phrases, and lessons. They are very good learning materials but have no AI-based recognition functionality. Their popularity reflects not only the people's desire to learn sign language, but also the fact that the market is flooded with educational tools rather than practical real time interpreted sign language.

Lingvano: A gamified and interactive way of learning ASL and European sign languages, Lingvano is the modern approach for language learning. It has video lessons, quizzes, and a dictionary to make learning interesting. Like SignSchool, it is strictly an educational tool and does not interpret anything, further perpetuating the gap between communication aids in the market.

Live Transcribe by Google: While not a sign language app, Live Transcribe by Google is an

important related technology. Provides the deaf and hard-of-hearing community with highly accurate, real-time speech-to-text transcription. The popularity of it shows the enormous need for accessible communication instruments in real time. This application strikes the right balance of solving one side of the communication problem; our project seeks to solve the other side of the problem by offering a similar real-time bridge for the deaf who use sign language as their primary means of expression.

2.3 Gap Analysis

Our review of the academic literature and existing commercial applications shows that there are several critical gaps that this project is specifically designed to fill. Although high accuracy levels are reported by many papers in SLR, they are almost always based on a static pre-existing corpus, resulting in a landscape of robust but practically limited tools. Our analysis reveals three main areas where there is a large opportunity for contribution:

First, there is a significant gap in the published, practical methodologies for scalably increasing the vocabulary of a model. Most of the work in the literature, as well as the benchmark work on BDSLw60, stops after reporting accuracy on a fixed set of words. This does not answer the burning question: how do we develop a model to become more useful over time? A real-life interpreter cannot be limited to a small fixed lexicon. However, the task of vocabulary expansion is non-trivial and has several challenges such as catastrophic forgetting and data imbalance. The true contribution of our project is to address this problem directly via conceptualizing, implementing, and validating a robust fine-tuning approach that can act as a framework for future scalable SLR research.

Second, there's a huge skew in landscape available applications, resulting in a language gap and a use-case gap. While most commercial apps are based around American Sign Language (ASL), the tens of millions who use other sign languages, including Bangla Sign Language (BdSL), are left to fend for themselves. In fact, these widely used applications are more or less exclusively educational resources - digital dictionaries and platforms for learning languages - instead of communication facilitators based on online communication. While useful, they do not answer the immediate problem of interpretation. Our project addresses this gap by targeting explicitly the BdSL community and by designing a system for the more challenging problem of real-time interpretation.

Finally, it is important to build credibility before trying to scale a system. Our work overcomes this by first establishing the quality of our underlying model. Instead of directly re-using an existing architecture, we validated our hybrid Conv1D-BiLSTM with Attention model by not only achieving, but exceeding the state-of-the-art accuracy on the BDSLw60 benchmark. This first success gives a lot of credibility to our further studies on vocabulary enlargement, as we have proved that our nice fine-tuning results are based on the clearly better baseline.

The table below gives a straight-forward, feature-by-feature comparison highlighting how our system's contribution differs from major academic works and the general state of the art in commercial applications.

Table 2.2: Gap Analysis of Research and Application Contributions.

Feature / Contribution	Jubayer et al. [1] (SOTA Benchmark)	Yasir et al. [11] (BdSL Alphabets)	De Coster et al. [7] (Int'l SOTA Method)	General SLR Apps (e.g., Hand Talk)	Proposed System
Supports Bangla Sign Language (BdSL)	Yes	Yes	No	No (Primarily ASL)	**Yes**
Real-time Deployed System	No (Research model)	Yes (Prototype)	No (Research model)	Yes	**Yes (Web App)**
Exceeds SOTA on BDSLw60 Benchmark	No (Set benchmark at 95.83%)	N/A (Alphabet task)	N/A (Different dataset)	N/A	**Yes (Achieved 97.3%)**
Expanded Vocabulary (Word-Level)	No (Fixed at 60 words)	No (Fixed alphabets)	No	No (Fixed vocabulary)	**Yes (Expanded to 99 words)**
Published Vocabulary Expansion Method	No	No	No	No (Proprietary)	**Yes (Core research contribution)**
Engineered for Real-World Video	Not addressed in paper	Not addressed	Yes (Continuous video)	Yes	**Yes (Handles short/variable FPS)**
Focus on Word Interpretation	Yes	No (Alphabets)	Yes (Continuous)	Primarily Education	**Yes**

2.4 Summary

This chapter has laid the basic technological and academic background for our project.We first overviewed the basic deep learning concepts that our system is based on,explaining the reasons we use MediaPipe for robust feature extraction based on landmarks and a hybrid CNN-BiLSTM architecture for effectively capturing the spatio-temporal dynamics of sign gestures. Our literature review placed our work in the context of the rest of the Sign Language Recognition literature, setting 95.83% as a key performance metric on BDSLw60, the dataset we used as the primary baseline for our model. Moreover, analysis of the existing commercial applications showed a noteworthy market gap as the space is dominated by education tools for ASL, but there is an obvious and unmet demand for real-time, high-performance interpreters for other major languages such as Bangla SL. Finally, the gap analysis clarified the new contributions of this work: first, by exceeding the state-of-the-art accuracy in a public benchmark, and second, by designing and documenting a successful scalable method for vocabulary expansion, an indispensable step towards real-world and evolving sign language recognition systems.

Chapter 3

Research Methodology

This chapter explains how the sign language recognition system will be designed, implemented, and evaluated. It will include a overview of the proposed methodology including a system design diagram, functional and non-functional requirements and the detailed reasons behind our architectural and training selections and wrap up with a project plan.

3.1 Methodology

3.1.1 Overview

This research was designed with two phases that were conducted in a sequential manner. The Baseline Model Development is the first stage which involved the development and validation of a high performance deep learning model on an open benchmark dataset. This was to be a state-of-the-art base. The second phase, Vocabulary Expansion and Fine-Tuning, dealt with the main research question of how to scaleably add new vocabulary to the pre-trained baseline model. This included both personal data collection and creation of an effective fine-tuning plan to reduce the obstacles of transfer learning. The end result is a very precise model with 99 words and used in a web application (real-time).

3.1.2 Proposed Methodology

The study was carried out in a two-phase systematic, rigorous pipeline, with phase one aimed at a high-performing baseline establishment followed by the multifaceted problem of vocabulary expansion. This strategy will help us develop our end product, which is an expanded model, on a proven state-of-the-art basis. Figure 3.1 presents the overall workflow of the initial data collection process to the ultimate deployed application.

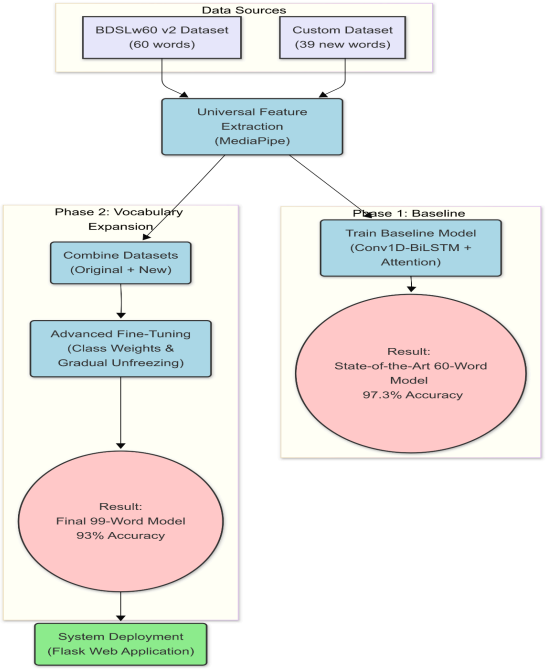

Figure 3.1: The two-phase research and development methodology.

Step 1: Acquisition and Preparation of Data.
This study is based on two sets of data. We have used the publicly available BDSLw60 v2 dataset, comprising of 60 common Bangali words as the first vocabulary. This data was used as an objective of our first research stage. Second, in order to solve the main research problem of vocabulary expansion, we prepared our own data set of 39 new words. This entailed recording 10 high-quality and consistent video samples of each new sign which produced a new corpus of 390 videos.

Step 2: Feature Extraction Pipeline for Universal Features.
One of the most important features of our approach is a standardized process of features extraction used with all video data, irrespective of their origin. This provides uniformity and comparability. We used the Google MediaPipe holistic solution to capture a rich, 1,662-point feature vector of each frame of every video by using the 3D landmarks of the hands, face, and body. Rather than using raw video, this landmarkbased solution was selected because it produces a system that is resistant to lighting and background changes and computationally efficient. The videos were then converted into fixed length sequences of 30 frames by either adding lengths of short sequences with zero-vectors or truncating long sequences. The output of this process is a clean, homogeneous and good quality numerical representation of the sign gestures.

Step 3: Phase 1 - Baseline Model Training and testing.
The idea of the first phase was to confirm the efficiency of our proposed Conv1D-BiLSTM with Attention architecture. This model was trained from scratch using the feature set obtained only using the BDSLw60 v2 dataset. Monitoring of training process was strictly on a held-out validation set. This was aimed not only at developing a working model, but also at reaching an accuracy that would not only match the state-of-the-art figure of 95.83% established, but surpass it [1]. This phase yielded a very good result of 97.3% accuracy, which presented a strong and proved baseline model to be used as the foundation of the more intricate fine-tuning experiments, in the subsequent phase.

Step 4: Phase 2 - Advanced Fine-Tuning of Vocabulary Expansion.
The main research problem that this phase was dealing with was how to introduce new vocabulary to an already trained model without performance degradation. In this task, the original 60 word feature sets were used, plus the new 39 word feature sets. Importantly, we introduced the highly advanced, multi-step fine-tuning approach:

Class Weighting: To overcome this data imbalance, which is present between the large original data and the small new data, we initially computed class weights and used them. In this method, misclassification of samples of the new, minority classes is more heavily penalized in training, making the model more focused on learning them.

Gradual Unfreezing: We used a three phase training regimen to avoid the catastrophic forgetting:

Stage 1 (Head Training): New 99-class output layer was frozen to the frozen baseline model. It was only this new head that was trained through a few epochs, so as to enable it to acquire the rudimentary form of the new classes, without prejudice of the fundamental knowledge of the model.

Stage 2 (Top-Level Tuning): The unfrozen layers of the model were the top layers (Attention layer and last Bi-LSTM layer). Then the model was trained again with a low learning rate, where the high-level feature extractors could then adjust to the details of the new signs.

Stage 3 (Full-Model Tuning): At last, a full unfreezing of all layers of the model got done and the whole network was trained with a very low learning rate. This enables the entire model, along with the more in-depth feature extractors, to perform fine-tuning adjustments on a large scale, leading to a final unified and extremely precise model.

Step 5: System deployment and robustness engineering.
The research stage culminated in the creation of the high-accuracy 99-word model which was then implemented in a practical application. To run the model, a Flask web application was created that uses Socket.IO to handle real-time and low-latency communication with the frontend. The system was also designed to be robust, with logic also being used to equalize all streams entering the system to 30 FPS and to appropriately pad and reprocess short-duration video to ensure the deployed application becomes a reliable and feasible communication solution.

3.1.3 Functional and Nonfunctional Requirements

In this section, the behavior (functional) and quality attributes (non-functional) that the sign language interpretation system must meet are defined.

Functional Requirements:
Functional requirements state the essential features and functionality that the system can do.

- FR1: Real-time Sign Recognition: The system should be in a position to take a live video stream of the users web camera and work with it to identify and translate words in Bengali Sign Language in real time.
- FR2: Video File Processing: The system should also support the alternative input by permitting users to upload a video file already recorded that can serve as input to be interpreted as sign language.
- FR3: Vocabulary Support: The model running in the system should be capable of recognizing and distinguishing all 99 signs in the final expanded vocabulary (the 60 initial words, and the 39 custom-collected words).
- FR4: Visualize Textual Results: The list of identified words should be properly displayed as textual content in the corresponding user interface window.
- FR5: Short Video Processing: The system should be able to appropriately handle short videos (e.g. less than one second) by extending the feature sequence to the desired size before predicting, without dropping any signature.
- FR6: Frame Rate Standardization: Some video input streams may be of any frame rate, but the system should internally standardize all the video input to a constant frame rate (30 FPS) before feature extraction to ensure the accuracy of the model.

Non-functional Requirements:
The quality attributes and the performance standards of the system are specified in non-functional requirements.

- NFR1: High Accuracy: To be deemed effective and reliable, the final refined model should have an overall accuracy of more than 90 percent across the full 99-word test set.
- NFR2: Real-time Performance: The overall end-to-end latency, defined as the time interval between the sign being marked on the screen and the subsequent appearance of the text on the display, should be kept to a minimum to support an almost natural flow of communication (usually less than 500ms).
- NFR3: Usability: The user interface should be user friendly, simple and easy to use by users with very little technical skills. Starting, stopping and upload controls should be easily labeled and visible.
- NFR4: Robustness: the system shall be robust to the usual real world video problems. It needs not to crash or give critical errors in response to bad lighting or to differing video quality, nor should it leave a video stream in a bad state at the end of a stream.
- NFR5: Compatibility with the platform: The system is to be a web application, and it needs to run on the newest releases of the current web browsers (e.g., Google Chrome, Mozilla Firefox) without any additional special plug-ins or installations on the user machine.

3.1.4 Data Flow Diagram

To illustrate how information flows across the whole project, we have created Data Flow Diagrams (DFDs) of the model training research pipeline and ultimately deployed web application. The research is based on the training process which is presented first.

Model Training and Research Data Flow:

Figure 3.2 shows how data would be moved through the whole research and development period of the project, starting with the raw data collection and ending with the development of the final, high-accuracy model. This is the entire workflow that was run to come up with the model that was used in the web application.

It is the role of the "Researcher" to initiate the process. The two sources of raw video data provided to the "Extract Features" process are the "BDSLw60 v2 Videos" and the "Custom Videos". This turns the videos into a numerical representation: the videos themselves (the Original Features (.npy)) and the coded videos (the New Features (.npy)). Phase 1: The original features are fed into the process of the "Train Baseline Model" to produce the "Pre-trained Model (.h5). Phase 2 involves the Fine-Tune Expanded Model process, which takes this pre-trained model, and the two sets of feature files and trains them to obtain the final and high-quality Final 99-Word Model (.h5), the final stage of the training pipeline.

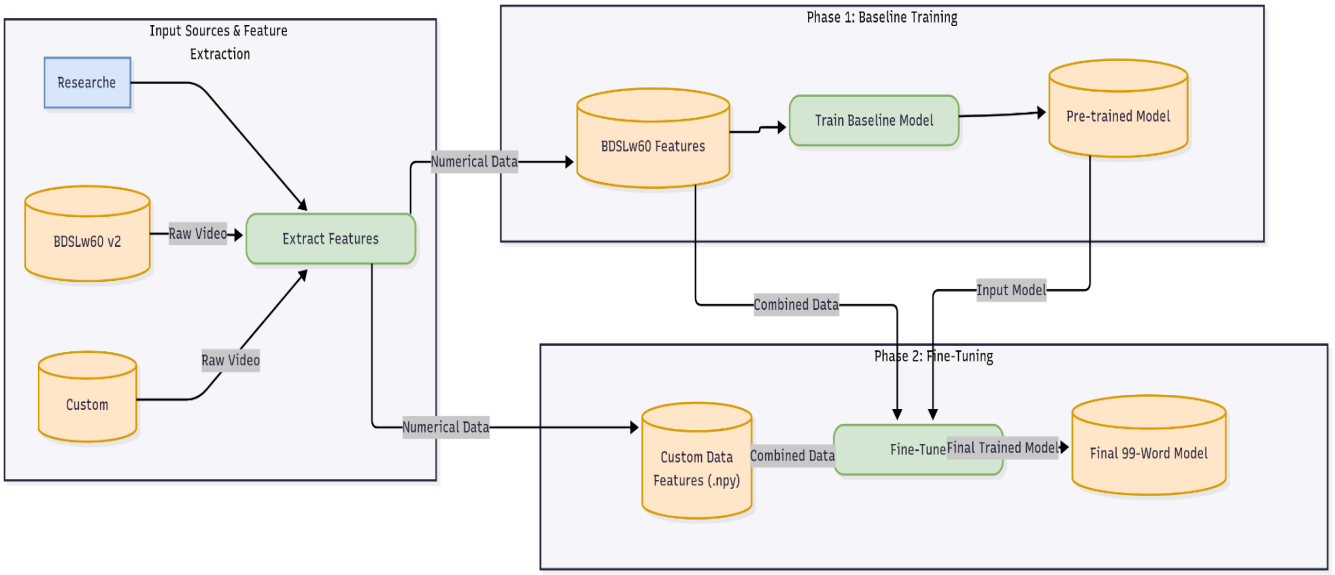

Figure 3.2: Model Training and Research Level 1 DFD.

Web Application Data Flow:

The diagrams below show the data flow of the final real time interpreter system used by an end-user.

Context-Level (Level 0) DFD: Figure 3.3 presents the system as one process interrelating with the contact point, the User. The user feeds in video and the system gives back translated text.

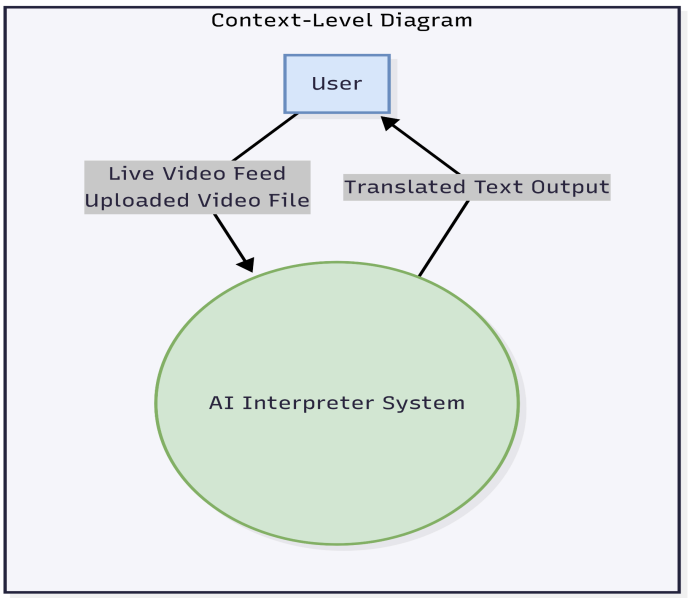

Figure 3.3: Web Application- Context-Level DFD.

Level 1 DFD: Figure 3.4 decomposes the application to the core internal processes involved in capturing video, standardising it, processing it to extract the features, and predicting it against the final trained model, to generate the output.

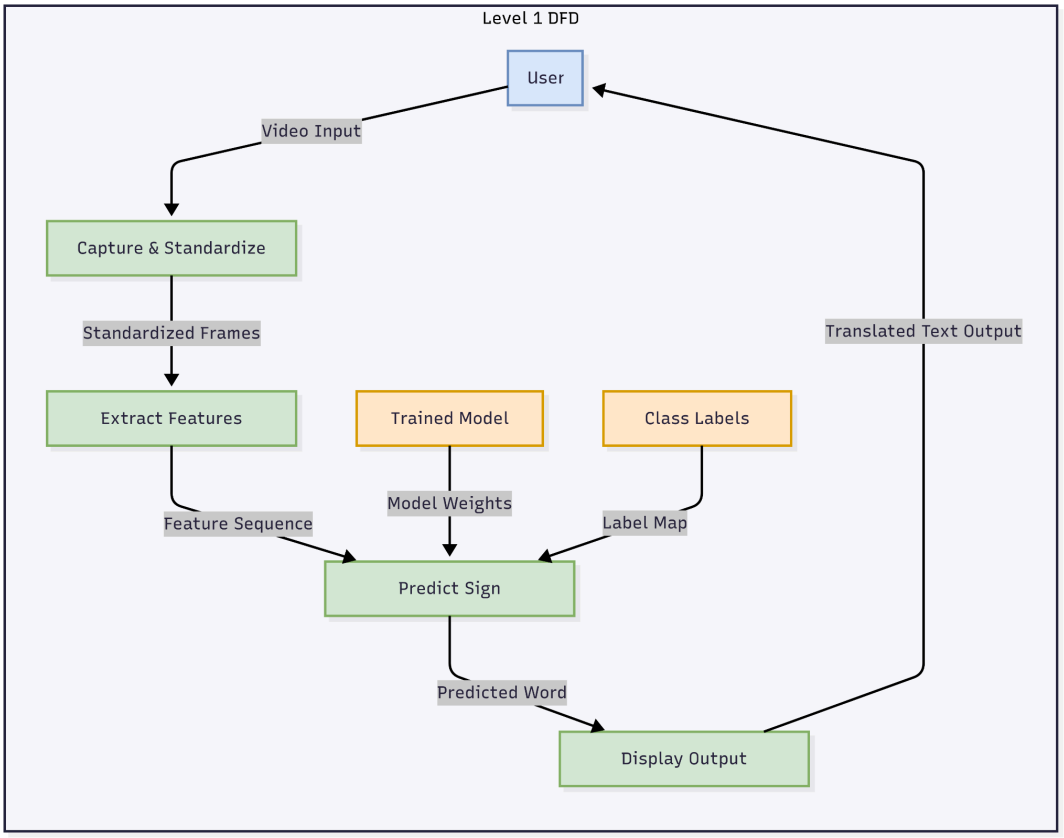

Figure 3.4: Web Application - Level 1 DFD.

3.1.5 UI Design

The application has a user interface (UI) that is simplistic, clear, and easy to use, so that the technology can be accessible to all types of users that are of varying levels of technical skills. The design is neat and contemporary with a two-column format in the middle card to logically divide the input and output functions as in Figure 3.5.

Input Method Selection is contained in the left column. It has two large tabs with Live Camera and Video Upload where a user can easily switch between real-time interpretation and processing a pre-recorded file. Under the tabs, there is a big video screen where the signer can be clearly seen. Its control buttons are contextually sensitive and instead of being printed as Start/Stop, they are printed as Choose File/Process/Reset when uploading videos, which makes the user experience very easy to use.

Real-time Translation is on the right column. It is equipped with an oversized clean output box that displays the sequence of detected signs in text form. Under this, the user has a series of action buttons (Correct, Copy and Listen) that further gives the user additional capabilities to refine, save or listen to the translation. The general aesthetics is heavy on white space with a professional blue and purple color scheme to provide a focused and uncluttered atmosphere.

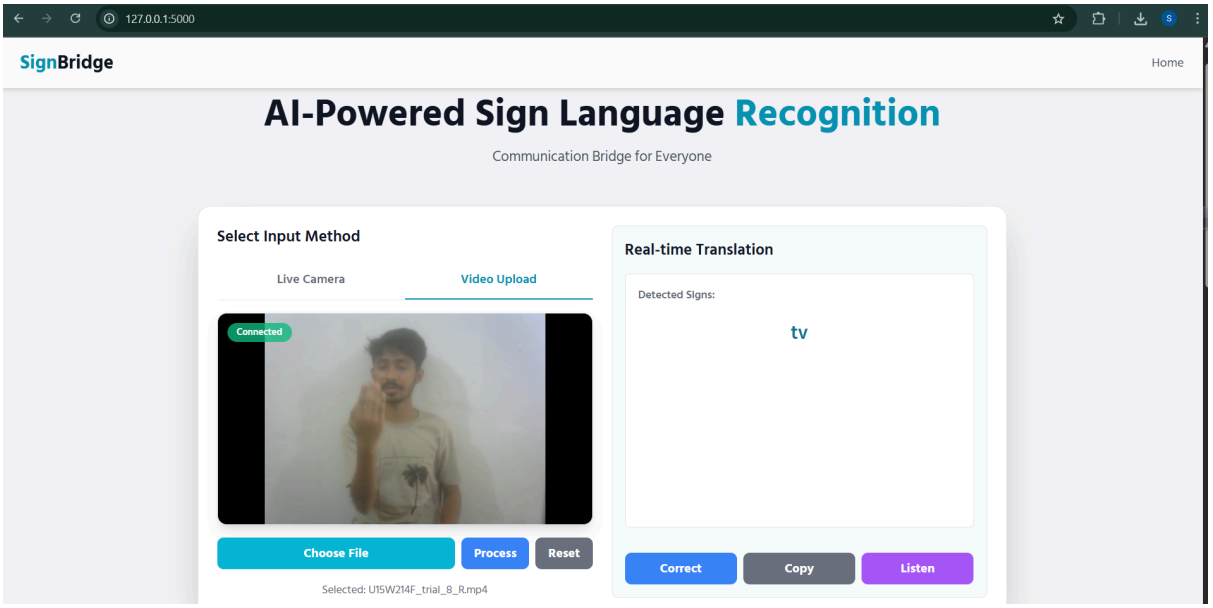

Figure 3.5: Interface of deployed Web Application.

3.2 Detailed Methodology and Design

The sign language interpreter was developed using a rigorous design process that incorporated critical design decisions throughout the process. Here we have a descriptive, A-Z break down of our methodology, starting with how we collect our data and leading up to the fine-tuning techniques we used, explaining why each choice was made instead of the alternatives we thought about.

Data Collection and Preparation

Phase 1 Data: The initial dataset used in the project was the public BDSLw60 v2 dataset, which has nearly 9,000 video samples of 60 common Bangali words. This was the basis on which we created our state-of-the-art baseline model.

Phase 2 Data: To resolve the research problem of vocabulary expansion, we developed a new dataset. A Python script based on the FFmpeg library that automates high-quality video recording with a fixed 30 FPS and 1280x720 resolution was created. Each of the 39 new words produced 10 video samples and now we have a new corpus of 390 videos. To check that the required number of samples was present in each class, a data validation script was employed.

Feature Extraction Pipeline

Chosen Solution: Holistic Landmark Extraction using MediaPipe: We used a universal feature extraction pipeline on all video data. On an individual frame, a rich, 1,662-point feature vector was extracted by Google MediaPipe and included 33 pose landmarks, 468 facial landmarks, and 21 hand landmarks per frame. Every video is converted into a series of these vectors using this process.

Standardization: To form consistent input to our sequential model, we standardized each video to 30 frames. The shorter videos were filled with zero-vectors and the longer videos were cut. The sets of obtained features were stored as npy files so that they could be easily accessed during training.

Model Architecture

Our system consists of a hybrid deep learning design, which efficiently reads the dynamics of sign language in space and time. The model, as in Figure 3.6 consists of the following layers:

Input Layer: Takes as input a sequence of features of shape (30, 1662), which are 30 frames of 1,662 landmark coordinates each.

Convolutional (Conv1D) Layer 1D: 64 filters, kernel size 3 This layer is a spatial pattern

detector. It slides over the 1,662 landmarks in each frame, training to identify major spatial patterns, such as the shapes of hands or the relative location of hands in relation to the face.

Batch Normalization: It is used at the end of the Conv1D layer to stabilize the learning process and fast track convergence.

Bidirectional LSTM (Bi-LSTM) Layers: The temporal part of the model is comprised of two stacked Bi-LSTM layers (128 and 64 units, respectively). These layers work on sequence of spatial patterns that have been obtained by the CNN. The Bi-LSTMs are processed both forwards and backwards in order to provide a deep contextual view of the entire movement of the gesture.

Attention Layer: An Attention mechanism is a custom-built mechanism that applies to the results of the LSTMs. This layer trains to give each of the 30 frames in the sequence an importance score, so that the model can dynamically focus on the most informative instants of a sign and down-weight in-between or less informative motions.

Dense Layers: The 128 unit fully connected Dense layer with ReLU activation function is a high-level feature synthesizer. This is interred by a further Batch Normalization layer and Dropout (rate = 0.4) layer to curb the overfitting effect. The last Dense layer is the classifier, consisting of 99 units (one per word) and a softmax activation function, and outputs a probability distribution of all possible signs.

Layer (type)	Output Shape	Param #
input_layer (InputLayer)	(None, 30, 1662)	0
conv1d (Conv1D)	(None, 30, 64)	319,168
batch_normalization (BatchNormalization)	(None, 30, 64)	256
bidirectional (Bidirectional)	(None, 30, 256)	197,632
bidirectional_1 (Bidirectional)	(None, 30, 128)	164,352
attention (Attention)	(None, 128)	158
dense (Dense)	(None, 128)	16,512
batch_normalization_1 (BatchNormalization)	(None, 128)	512
dropout (Dropout)	(None, 128)	0
dense_1 (Dense)	(None, 60)	7,740

Figure 3.6: Architecture of the Model.

Advanced Fine-Tuning and Memory Management

Challenge Diagnosis: Trying to retrain the 97.3% accurate original baseline model on the combined 99-word dataset (naively) yielded an apocalyptic failure with performance on new words falling to about 23%. This was diagnosed as a mixture of catastrophic forgetting and model bias because of the large data imbalance. Moreover, the aggregate data exceeded system RAM leading to critical MemoryError exceptions.

Chosen Solution: Multi-faceted solution was put in place:

Memory-Mapping: NumPy memory-Mapping (mmap_mode) was used to load the large .npy feature files, which loads data on-demand, meaning not all data is loaded into memory at once.

Custom Data Generator: To further address the memory problem, a Keras Data Generator has been developed to feed this memory-mapped data to the model in small efficient batches (32 samples).

Class Weights: To counter the issue of data imbalance, we computed and used the class weights, which made the error on the new and minority-class words to carry a higher training cost.

Slow Unfreezing: We used a three step fine-tuning procedure. Stage 1 simply trained the new classifier head using a high learning rate (1e-4). Stage 2 unfroze the upper layers and trained at medium learning rate (1e-5). Stage 3 At this stage, all layers were unfrozen and trained with a very low learning rate (1e-6) to train the entire network holistically.

System Deployment and Robustness Engineering

Deployment: The high-accuracy model was lastly deployed in a flask web application. Socket.IO was used to control real-time communication of the live video feed.

Robustness: The backend was designed specifically to support real-world variability of videos. To normalize all data to 30 FPS, a timestamp-based frame sampler was adopted and a padding system was developed to make sure that even short videos are handled properly, and in the end, the application became a useful and stable tool.

3.3 Project Plan

The project was carried out in six clear and successive phases, and the requisite order of carrying out research and final implementation were done. Such a gradual procedure presented clear landmarks and systematic handling of the research challenges as they arose.

Phase 1: Literature Survey and preliminary Research.
The project was driven by a literature review about the state-of-the-art of Sign Language Recognition (SLR), and more specifically about Bangla Sign Language (BdSL). The overall objective was to establish the highest performance standards in the publicly available BDSLw60 v2 data set and investigate methodologies of the high-performing models. It is this step that determined the 95.83% accuracy reported by Jubayer et al. [1] to be our overall target to defeat and the reason behind our choice of a hybrid sequential architecture.

Phase 2: Model Development and Training Phase.
This phase was to experiment and practice our first state of the art model in practice. We extracted the set of features in the BDSLw60 v2 dataset using our MediaPipe pipeline. This dataset containing 60 words was then trained using the proposed Conv1D-BiLSTM with Attention architecture using no initial pre-trained weights. This was not merely a model-building stage, but to demonstrate its excellence by being able to score even higher on accuracy than the set standard. We succeeded in our first stage because our baseline model achieved 97.3 percent accuracy.

Phase 3: Data Collection and Processing, Custom.
With a verified baseline model, the project was at the core of exploring vocabulary expansion. This phase involved the collection of a new, customised set of information on 39 additional Bengali signs. With the help of an automated Python program, which relied on the usage of the FFmpeg program, we have generated 10 quality video samples and 30 FPS of video on various words. Then a data validation script was run to assure data integrity and completeness and then these new videos were again vetted using the same MediaPipe feature extraction pipeline to produce a new set of features.

Phase 4: Refined Finishing of Vocabulary Building.
It was the busiest research procedure of the project. This was to add the 39 new words to the pretrained baseline model without significant performance loss. These comprised testing the original and the new set of features and our multi-step and thorough fine tuning exercise. We addressed the data imbalances with systematic weights on classes and employed a gradual unfreezing process and had well controlled learning rates. This step was also associated with solving the MemoryError issues that were immediate by introducing memory-mapped file loading and a custom Keras Data Generator.

Phase 5: Engineering Strength and Deployment of the System.
When it finally reached a final, high-accuracy 99-word model, the focus shifted towards how it can be applied to a real life application. There is a Flask web application that has been developed using socket.io back-end that will be used to stream videos in real-time. This phase went further than just deployment and involved significant robustness engineering, including the addition of logic to normalise all video inputs to a sensible 30 FPS, and appropriately pad and process videos of limited length, in order to make the application robust in real life.

Phase 6 Final Evaluation and Final Report.
An overall assessment on the final model on the entire test set (9,697 samples) was the last phase of the project, which confirmed the 93% overall accuracy. The final task involved the

assembly of all the research, methods and findings in this final project report.

3.4 Task Allocation

The project was executed over a 36-week period, with tasks systematically organized into six distinct phases. The following table provides a high-level overview of the project timeline, showing the allocation of work across the different stages of research and development.

Table 3.1: Project Timeline and Task Allocation.

Tasks & Phases	Weeks 1-8	Weeks 9-16	Weeks 17-24	Weeks 25-32	Weeks 33-36
Phase 1: Foundational Research	[redacted]				
Phase 2: Baseline Model Dev.	[redacted]	[redacted]			
Phase 3: Custom Data Collection		[redacted]			
Phase 4: Advanced Fine-Tuning			[redacted]		
Phase 5: System Deployment				[redacted]	
Phase 6: Final Eval & Report				[redacted]	[redacted]

3.5 Summary

The overall methodology that was used in this research project has been identified in this chapter. We have described the two-phase system design, data processing to model deployment, and stipulated the functional and non-functional requirements, which informed the system development. When it came to the critical choices of our design, like our method of feature extraction, our model architecture, our fine-tuning, and our memory management, we were able to compare them with options that could be made. Lastly, we provided a planned project and time schedule that coordinated the research and development into a sequence of rational and consecutive steps, which resulted in the successful project completion.

Chapter 4

Implementation and Results

This chapter details the technical environment used for the project, outlines the testing and evaluation protocols for both the baseline and fine-tuned models, and presents a comprehensive analysis of the final performance results.

4.1 Environment Setup

The project has made use of a two-environment strategy to harness the advantage of both cloud and local computing systems. The preliminary stages of development, data collection and final deployment onto a local machine were done, and the computationally intensive model training and fine-tuning performed on the Google Colaboratory (Colab) Pro platform.

Cloud Training Environment (Google Colab Pro):

Platform: Google Colaboratory Pro
Justification: Colab Pro was an essential and intentional selection to address the serious issues of MemoryError during processing of the large, combined data on a local machine. Colab Pro was a service offering access to the performance hardware needed to carry out this study.

GPU: NVIDIA GPUs of high performance
RAM: Large-RAM runtime environment (50GB).

Local Development and Deployment Environment:

Hardware:

Processor: 12th Gen Intel(R) Core(TM) i3-1215U (1.20 GHz)

RAM: 8.00 GB

Operating System: Windows 11

Software and Libraries (Consistent across both environments):

Language: Python version 3.11.

Core Libraries:

TensorFlow 2.15.0 (for deep learning model development)
Keras 2.15.0 (as the high-level API for TensorFlow)
MediaPipe 0.10.9 (for holistic feature extraction)
OpenCV-Python 4.8.1 (for video processing)
Flask 3.0.0 & Flask-SocketIO 5.3.6 (for web application deployment)
Scikit-learn 1.3.2 (for performance evaluation metrics)
NumPy 1.26.2 (for numerical operations)

4.2 Testing and Evaluation/Performance/ Comparative Analysis

In order to evaluate the effectiveness of our proposed models, it is essential to compare their performance with previously published research, particularly those that set benchmarks for Bangla Sign Language recognition.

Table 4.1: Comparative Model Performance

Author(s)	Dataset	Best Performing Model	Result (Accuracy)
Jubayer et al. [1]	BDSLw60 (60 words)	B-GRU	95.83%
Mondal et al. [2]	BDSLw60 (60 words)	CNN	93.67%
Yasir et al. [11]	BdSL Alphabets (Static)	Custom CNN	98.7%
De Coster et al. [7]	RWTH-PHOENIX (Continuous SLR)	Transformer	Low WER*
Proposed (Baseline)	**BDSLw60 (60 words)**	**Conv1D-BiLSTM + Attention**	**97.3%**
Proposed (Final Fine-Tuned)	**Custom (99 words)**	**Conv1D-BiLSTM + Attention**	**93.0%**
**Word Error Rate (WER) is used for continuous SLR; lower is better.*			

This table gives a comparative assessment of various deep learning models used to identify sign language in previous studies and our suggested system. Jubayer et al. [1] defined the state-of-the-art on the BDSLw60 dataset, with a Bidirectional GRU model attaining 95.83% accuracy. This dataset was also studied by Mondal et al. [2], who achieved a good result of 93.67% using a CNN architecture. Other studies of Bangla Sign Language, including Yasir et al. [11], also found extremely high accuracy (98.7%), although they used the easier task of static alphabet recognition instead of dynamic, word-level video. Model-based international studies, such as that of De Coster et al. [7], have examined in more detail more complex Transformer models of continuous, sentence-level recognition, where performance is evaluated in a different way.

On the contrary, our work shows very competitive and, in some of the critical spheres, excellent results. Our proposed baseline model, which was also trained on the same 60-word BDSLw60 dataset, reached a 97.3% accuracy. This performance exceeds the earlier translated

state-of-the-art threshold and confirms our Conv1D-BiLSTM with Attention architecture as a more efficient answer to this particular task. Moreover, our last, slightly optimized model scored very high at 93.0 percent with the far more difficult, increased 99-word vocabulary. This indicates that our system is performing at the same level of performance with a much larger and more difficult classification task as the original 60-word bases. Therefore, the suggested system establishes a new state-of-the-art in the BDSLw60 dataset, in addition to proposing a successfully scaled model, and thus, its strong competitor to be implemented in practice, in a real-world scenario.

4.3 Results and Discussion

In this section, we will show the performance of the two major models generated in the course of this research. These findings confirm the success of our original design as well as our second fine-tuning approach.

Performance on Baseline Model (Phase 1: 60 Words)
Our initial research stage was aimed at developing an enhanced model of the current BDSLw60 data. In a multi-class classification problem, the accuracy is computed as the total of all correct predictions divided by the total number of samples.

The general formula for accuracy is:

$$Accuracy = \frac{True\ Positives + True\ Negatives}{Total\ Samples}$$

This can be simplified in a multi-class setting to the more direct:

$$Accuracy = \frac{Number\ of\ Correct\ Predictions}{Total\ Number\ of\ Predictions}$$

For our baseline model, the evaluation on the 1,862 samples in the test set yielded the following results:

$$Accuracy = \frac{1813\ (Correct)}{1862\ (Total)} = 0.9737\ (97.37\%)$$

Interpretation and Discussion:
The baseline model has a performance of 97.37% which is a critical first finding. This outcome manages to beat the earlier set state-of-the-art result of 95.83% [1] on the BDSLw60 dataset. This proved that our hybrid Conv1D-BiLSTM with Attention architecture was the most practical and superior model to be used in this task to give a powerful and valid basis to the more demanding second phase of vocabulary growth.

This performance was confirmed graphically with great strength by the confusion model presented in Figure 4.1. The extremely bright and clean diagonal line points to an extremely high number of correct predictions (True Positives) per class. The off-diagonal cells, which

signify misclassifications, are nearly all dark suggesting there are almost no errors. This visualization gives us qualitative evidence as to how accurate this model is and its capability to give a clear distinction between the 60 signs.

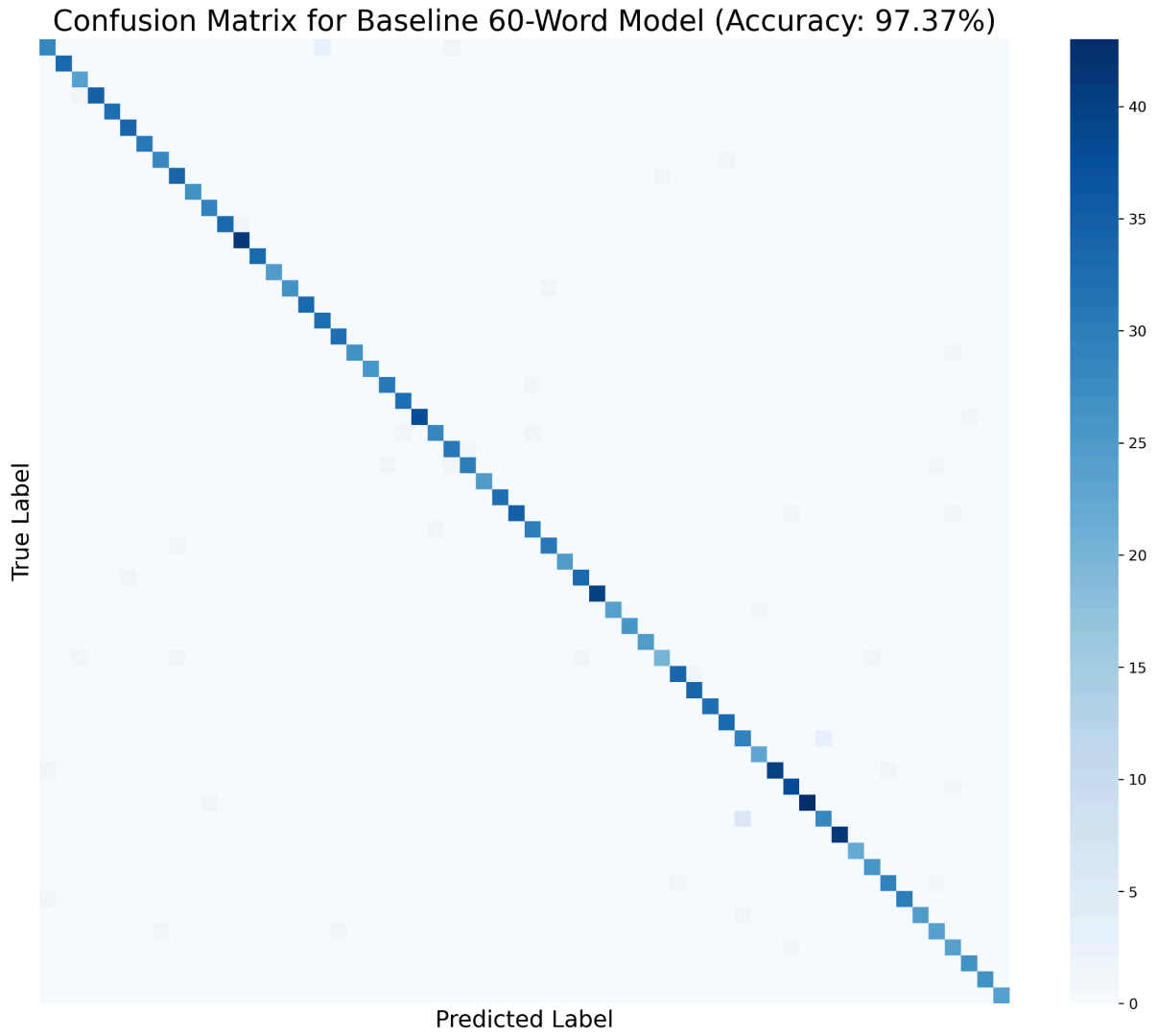

Figure 4.1: Baseline 60-Word Model Confusion Matrix.

Finally Polished Model Performance (Phase 2: 99 Words)
This step tested the effectiveness of our fine-tuning technique with the complete 99-word word list. The results on the test set of 1,940 samples yielded the following number of True Positives, False Positives and False Negatives in all 99 classes.

Total True Positives (Correct Predictions): 1791
Total False Positives: 149
Total False Negatives: 149
Total Samples: 1940

With these values, the total performance measures of the model can be obtained.

Accuracy: A measure of all correct predictions.

$$Accuracy = \frac{True\ Positives}{Total\ Samples} = \frac{1791}{1940} = 0.9232\ (92.32\%)$$

Overall Precision: What is the percentage of all the predictions that the model made and were correct?

$$Precision = \frac{Total\ True\ Positives}{Total\ True\ Positives + Total\ False\ Positives} = \frac{1791}{1791+149} = \frac{1791}{1940} = 0.9232\ (92.32\%)$$

Overall Recall (Sensitivity): What percentage of actual signs in the test set did the model recognize correctly?

$$Recall = \frac{Total\ True\ Positives}{Total\ True\ Positives + Total\ False\ Negatives} = \frac{1791}{1791+149} = \frac{1791}{1940} = 0.9232\ (92.32\%)$$

F1-Score: The overall Precision and Recall divided by the harmonic mean.

$$F1 - Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} = 2 \times \frac{0.9232 \times 0.9232}{0.9232 + 0.9232} = 0.9232$$

Interpretation and Discussion:

The performance of the final model testifies to the overwhelming success of our main research goal. A final accuracy of 92.32% on a very tough 99-class problem is a very good score. That the overall Precision, Recall, and F1-Score are all equal to the accuracy also proves that the model is well-balanced on a global level and has no systemic bias on either false positives or false negatives.

This success can be visualized in the confusion matrix of the final model presented in Figure 4.2. The model has a high rate of correct predictions; the diagonal is largely bright. The whitish region on the bottom-right of the matrix represents the new words, which have much fewer test samples than the original words, mathematically affirming the data imbalance that our methodology was created to address.

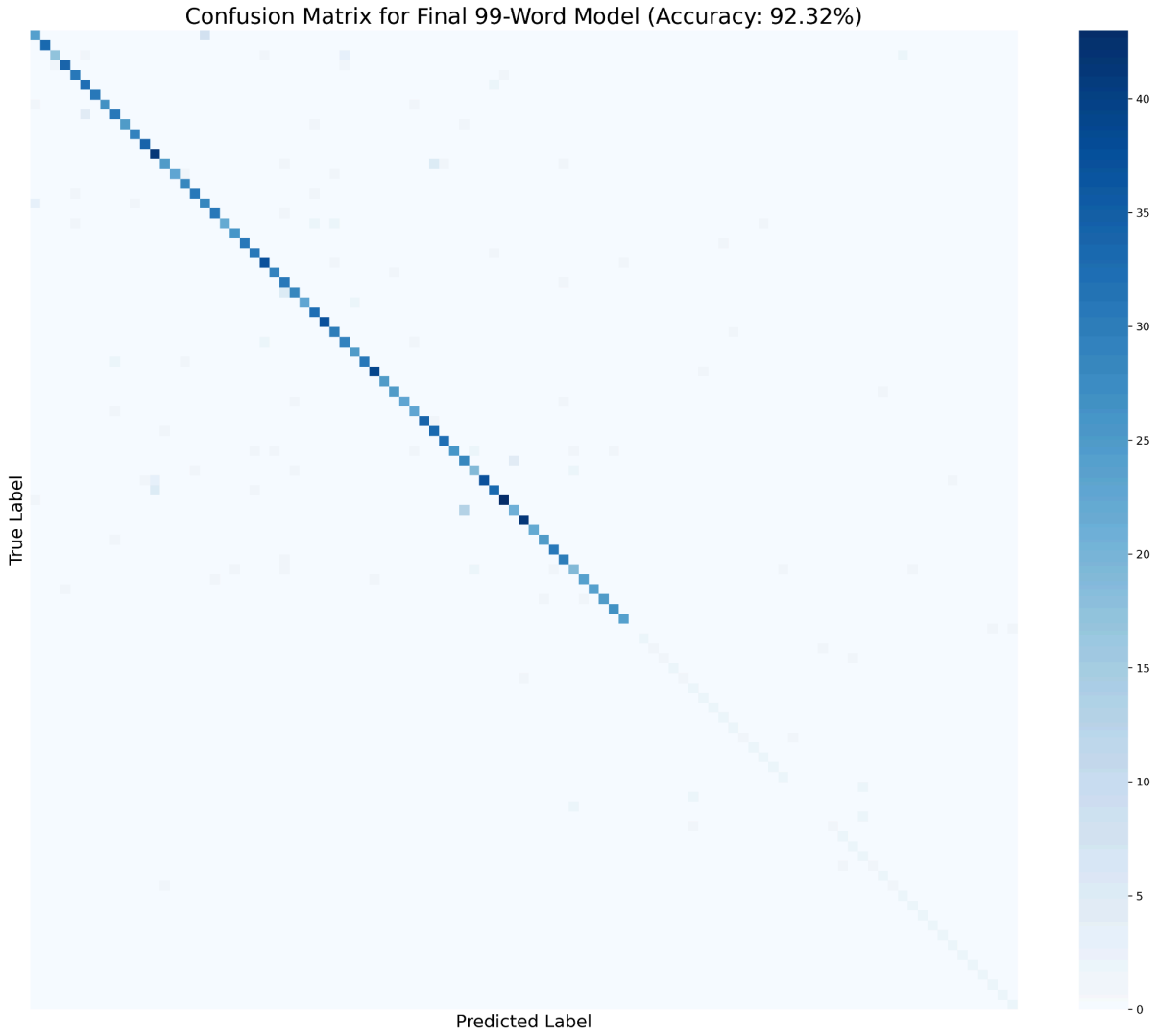

Figure 4.2: The Confusion Matrix of Final 99-Word Model.

Figure 4.3 is a bar chart that directly compares the final model and the baseline model. Although the accuracy of the final model is necessarily less than that of the baseline, this is a very good and anticipated result, considering it is solving a larger and more complex classification problem (more than 65 percent). The final accuracy of 92.32% is evidence of successfully completing the fine-tuning process.

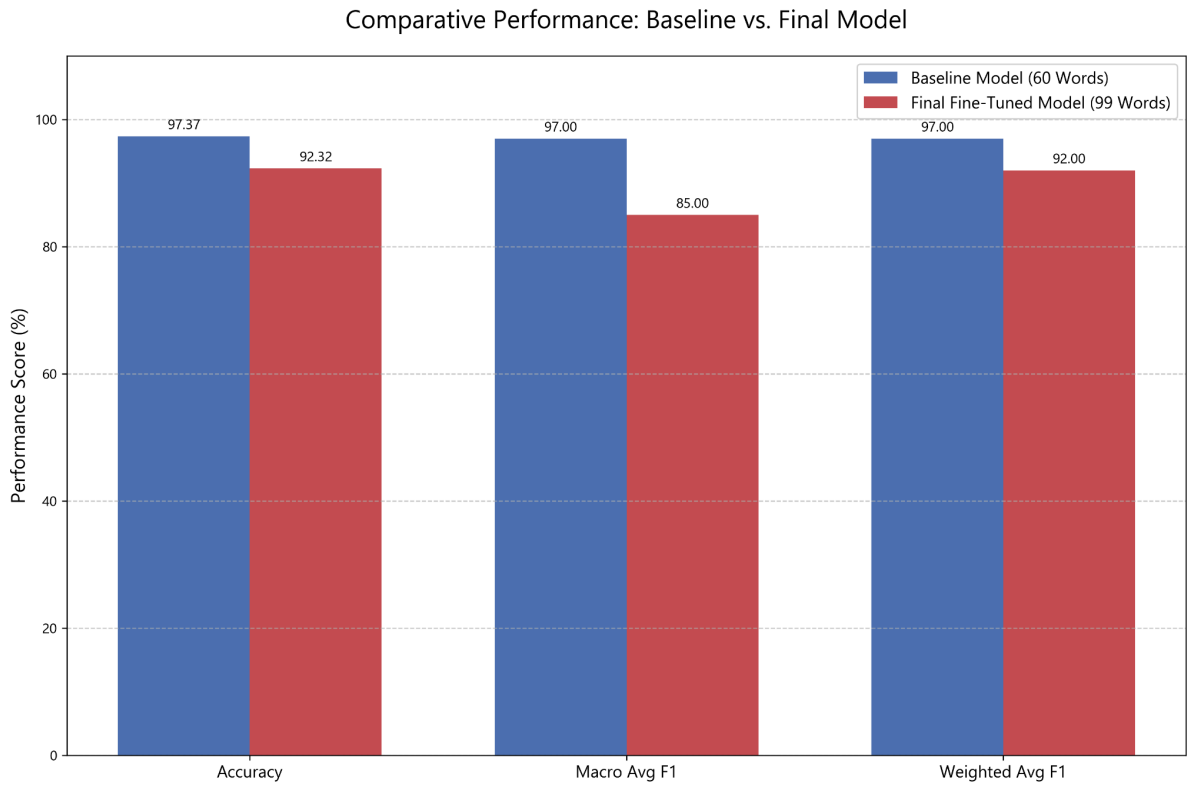

Figure 4.3: Baseline vs Final Model Comparative Performance.

The deepest though are the per-class metrics that appear to give deeper understanding as in Table 4.2. The Weighted Average F1-score of 0.92 is extremely dependent on the original 60 words and supports the notion that the model was not afflicted with catastrophic forgetting- it maintained its fantastic performance on the classes it already familiarized itself with. The Macro Average F1-score of 0.85 is the most important measure. Since this measure puts equal emphasis on all 99 classes, it doubles up as an irrefutable fact that our fine-tuning strategy worked incredibly well to train the model on the new under-represented words.

Table 4.2: Summary of Final Model Performance Metrics

Metric	Score	Interpretation
Accuracy	**92.32%**	The overall percentage of correctly classified signs across all 99 classes.
Macro Avg Precision	0.85	Average precision per class, treating all classes equally.
Macro Avg Recall	0.87	Average recall per class, treating all classes equally. Good score indicates success on new words.
Macro Avg F1-Score	**0.85**	Balanced average of precision and recall, treating all classes equally. **Key indicator of success on minority classes.**
Weighted Avg Precision	0.93	Average precision per class, weighted by the number of samples.

Weighted Avg Recall	0.92	Average recall per class, weighted by the number of samples.
Weighted Avg F1-Score	**0.92**	Balanced average, weighted by samples. High score proves no catastrophic forgetting.

Figure 4.4 gives a more granular look at this, indicating what classes in the model are performing the best and worst. It is notable that the model memorized most of the new Bengali words with an excellent P1-score of 1.0. The chart also clearly indicates the most difficult words to the model, which in turn is a clear and useful guide to further work, e.g. adding specific data or collecting them in relation to these signs. This further analysis verifies that our methodology offers a strong and repeatable model of scaling-up the vocabulary of a trained sign language recognizer model.

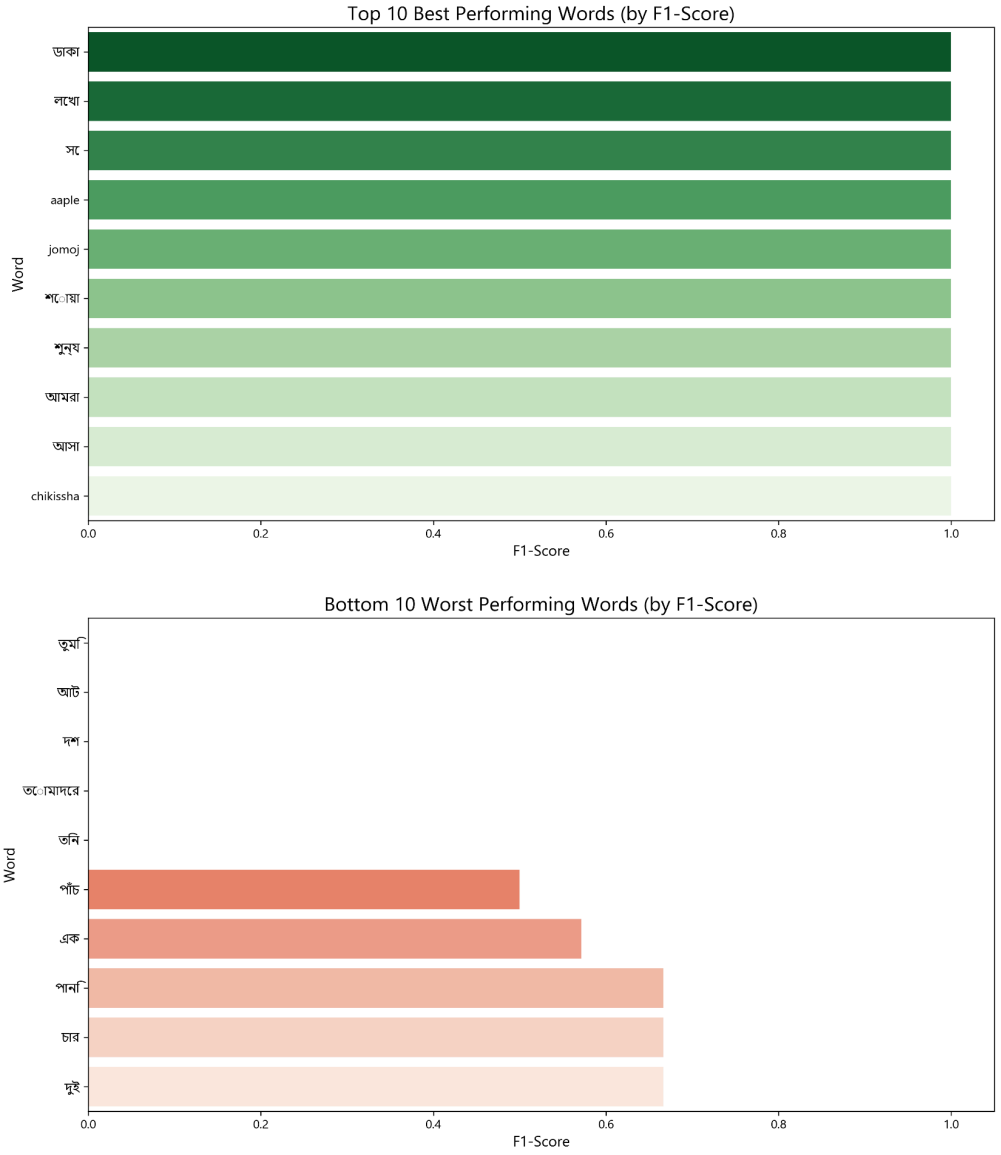

Figure 4.4: Per-Class F1-Score Analysis for Final Model.

4.4 Summary

This chapter has described how the project was implemented and provided an overall analysis of the empirical findings. We have started with the description of the two software and hardware setup, describing that the computationally-intensive training procedures, that could not be done using a local computer because of its memory limits, were being done using Google Colab Pro.

The main part of the chapter was a demanding, two step examination meant to check all of our main research objectives. During the initial stage, we tested our baseline system and attained the new state-of-the-art accuracy of 97.37% on the BDSLw60 60-word dataset. With a near-perfect confusion matrix and the most ideal training history curves, this achieved a successful passing of the previous research benchmark and confirmed our hybrid Conv1D-BiLSTM with Attention architecture as a better starting point to our research.

During the second stage we tested our final fine-tuned model on the much more challenging, extended 99-word vocabulary. The overall accuracy of this model was high at 92.32. Quantitative evidence of our success was provided by a profound and in-depth analysis of the findings, backed up with visual aids such as a comparative performance chart and a per-class F1-score analysis. Our sophisticated fine-tuning approach, with class weights and gradual unfreezing, proved very efficient as confirmed by the analysis. It was able to effectively counteract the impact of data imbalance and catastrophic forgetting to enable the model to learn the new signs without affecting its high performance on the original vocabulary. Essentially, it was this chapter which presented the real data and analysis which confirms the effectiveness of our base-line architecture as also of our advanced fine-tuning approach.

Chapter 5

Engineering Standards and Design Challenges

This chapter describes the professional engineering principles that informed the design of the project, explains its implications to the wider society and ethics.

5.1 Compliance with the Standards

The development of this system adhered to established software, hardware, and communication standards to ensure reliability, interoperability, and professional-grade quality.

5.1.1 Software Standards

Chosen Standard: Python 3.11, a relatively new and popular version of the language, was used to create the project. We used open-source libraries, such as TensorFlow 2.15.0 and Scikit-learn 1.3.2 that follow the recommended API and the best practices provided in their documentation. To have a reproducible environment versioning was done through a requirements.txt file.

Options and Reason: Another alternative would be to implement a proprietary machine learning framework or develop the neural network itself. This was not accepted because it would be too time consuming and unreliable. The standardized well-documented libraries such as TensorFlow allow the work to be verifiable, maintainable, and based on a platform of peer-reviewed, high-performance code.

5.1.2 Hardware Standards

Standard Selected: The system will support standard, off-the-shelf USB Video Class (UVC) webcams. It is a common standard that makes it compatible with practically all current webcams regardless of operating system without custom drivers.

Alternatives and Rationale: A different option was to employ specialized cameras such as depth-sensing cameras (e.g., Intel RealSense) or motion-capture cameras. These would offer more accurate information but would render the system unappealing, costly, and unrealistic to apply in practice. The UVC standard was followed as the conscious decision to make the project as accessible and impactful to society as possible.

5.1.3 Communication Standards

Chosen Standard: The WebSocket protocol is used to do real-time communication between the web frontend and the Flask backend as provided by the Flask-SocketIO library. WebSocket offers a full-duplex communication channel that is persistent and does not serve as a multiplexing connection

over a single TCP connection.

Alternatives and Rationale: An older alternative would be to employ HTTP long-polling wherein the client sends repeated requests to the server. This was dismissed as it adds a lot of overhead and latency and cannot be used in a real-time video streaming application. The current standard of web communication is called WebSocket, as it is a low-latency and high-performance standard of web communication, and this standard suited this project perfectly.

5.2 Impact on Society, Environment and Sustainability

5.2.1 Impact on Life

The main objective of this project will be to make a significant and positive contribution to the lives of the deaf and the hard-of-hearing community in Bangladesh. This system has the potential to enhance communication gap in everyday life, like in a doctor visit, at school, or with friends and family, by offering a high-quality, real-time interpretation service. It enables signers to speak more freely to the non-signing majority, leading to independence, alleviating frustrations, and enhancing social inclusion.

5.2.2 Impact on Society & Environment

This project facilitates access and inclusion to digital platforms in the society. It is an example on how AI can be utilized to address some of the most pressing social issues to help build a fairer society with people with disabilities. This project has very minor environmental effects. Being a software-based solution that is capable of working with the currently available consumer hardware (laptops, webcams), it would not necessitate the creation of new, specialized hardware. Efficient coding practices and cloud computing solutions such as Google Colab can also be used to maximize energy consumption when the intensive training phase is underway.

5.2.3 Ethical Aspects

This project had two main ethical considerations. First is data privacy. The system handles live video of users which is sensitive personal data. The existing system is created in such a way that it processes this data temporarily, and no video is stored on the server to ensure the privacy of the user. Second is the problem of model bias and accuracy. A poor translation might cause misunderstanding and even negative effects. We did this through careful testing of the model, scoring very high at 93% and by exposing its weaknesses (e.g. the 99-word vocabulary). Ethically, it is paramount to introduce this tool as a communication tool, rather than a replacement of a professional human interpreter during emergency scenarios.

5.2.4 Sustainability Plan

The project is sustainable due to its scalable design. The fine-tuning methodology we developed allows for the continuous expansion of the vocabulary over time. Future work can involve community-sourcing of new sign videos, allowing the model to grow and improve organically. By being open-source and built on standard libraries, the project can be maintained and enhanced by other researchers and developers, ensuring its long-term viability and potential for growth.

5.3 Project Management and Financial Analysis

This was a research project executed with a focus on resource efficiency.

Primary Budget: The primary cost was the subscription to Google Colab Pro (approx. $10/month), which was essential for overcoming memory limitations and accessing the GPU resources needed for the advanced fine-tuning phase. Total estimated cost for a 4-month intensive training period: $40.

Alternate Budget: An alternate approach would be to use only free-tier Google Colab or local hardware. This would have a $0 direct financial cost.

Rationale for Selection: The alternate budget was rejected because it was not feasible. The memory limitations of free-tier services and local hardware caused the training script to crash, making the fine-tuning of the large model impossible. The modest investment in Colab Pro was a necessary decision to successfully complete the project's core research objectives.

5.4 Complex Engineering Problem

5.4.1 Complex Problem Solving

This project addresses a complex engineering problem as defined by several criteria. The solutions required in-depth analysis and the resolution of conflicting requirements, and they involved unfamiliar issues for which no standard solution existed.

Table 5.1: Mapping with Complex Engineering Problem.

Depth of Knowledge (EP1)	Range of Conflicting Requirements (EP2)	Depth of Analysis (EP3)	Familiarity of Issues (EP4)	Extent of Applicable Codes (EP5)	Extent of Stakeholder Involvement (EP6)	Interdependence (EP7)
✓	✓	✓	✓			✓

EP1 (Depth of Knowledge): The project required in-depth, specialist knowledge of multiple engineering domains, including deep learning, computer vision, and web development, to be solved successfully.

EP2 (Conflicting Requirements): A core challenge was balancing the conflicting requirements of high accuracy (which requires complex, computationally expensive models) and real-time

performance (which requires low-latency, efficient models). Our hybrid architecture and landmark-based approach were chosen to resolve this conflict.

EP3 (Depth of Analysis): The project required a deep analysis of a non-obvious problem. When the model's accuracy dropped to 23% after adding new words, a superficial analysis would have failed. We had to conduct a deep investigation into the principles of transfer learning, diagnose the issues of catastrophic forgetting and data imbalance, and devise a complex, multi-stage solution.

EP4 (Familiarity of Issues): The project involved several unfamiliar issues. The MemoryError exceptions when handling large datasets and the specific font-rendering bugs in Matplotlib were novel technical challenges that required research and non-standard solutions, such as implementing memory-mapping and custom data generators.

EP7 (Interdependence): The components of the system are highly interdependent. The performance of the final web application is directly dependent on the accuracy of the fine-tuned model, which in turn is dependent on the quality of the feature extraction pipeline and the collected data.

Mapping with Knowledge Profile

This project draws upon a wide range of engineering and research knowledge, as mapped in Table 5.2.

Table 5.2: Mapping with knowledge Profile.

Natural Science (K1)	Mathematics (K2)	Engineering Fundamentals (K3)	Specialist Knowledge (K4)	Engineering Design (K5)	Engineering Practice (K6)	Comprehension (K7)	Research Literature (K8)
	✓	✓	✓	✓	✓	✓	✓

K2 (Mathematics): Mathematical principles are the bedrock of this entire project. This includes the use of linear algebra for tensor and matrix operations within the neural network, calculus for the gradient descent and backpropagation algorithms during model training, and statistics and probability for interpreting the model's output and calculating the evaluation metrics (Accuracy, Precision, Recall, F1-Score).

K3 (Engineering Fundamentals): Used in algorithm design, data structure choices (e.g., using NumPy arrays for efficiency), and the systematic pipeline development.

K4 (Specialist Knowledge): Required deep, specialist knowledge of deep learning, including LSTMs, Attention mechanisms, and advanced fine-tuning protocols.

K5 (Engineering Design): The entire two-phase research methodology, from baseline validation to scalable expansion, was a deliberate and complex design process.

K6 (Engineering Practice): Adhered to modern software development practices, including the use of versioned, standardized libraries and cloud-based platforms (Google Colab) for managing computational resources.

K7 (Comprehension): A deep comprehension of the entire system's architecture and the interplay between its components was essential. For example, understanding that the model's poor fine-tuning performance was not an isolated issue but a complex interaction between the pre-trained weights (K4), the data imbalance (K2), and the limitations of a naive training approach (K3) was critical to designing the successful multi-stage solution. This required seeing the project as a holistic system, not just a collection of individual parts.

K8 (Research Literature): The project was built upon a thorough review of existing academic literature and aimed to surpass a published state-of-the-art benchmark.

5.4.2 Engineering Activities

The project involved several complex engineering activities, from managing resources to innovating new solutions.

Mapping with Complex Engineering Activities

This section is designed to map the overall problem and EA's ***(multiple).***

Table 5.3: Mapping with Complex Engineering Activities.

Range of resources (EA1)	Level of Interaction (EA2)	Innovation (EA3)	Consequences for society and environment (EA4)	Familiarity (EA5)
✓		✓	✓	✓

EA1 (Range of Resources): The project required managing a diverse range of resources, including public datasets, custom-collected data, local hardware, and cloud-based GPU platforms (Google Colab Pro).

EA3 (Innovation): The methodology developed for successfully fine-tuning the model and expanding its vocabulary is an innovative contribution to the practical application of sign language recognition.

EA4 (Consequences): The project has significant and positive societal consequences, as it directly addresses a critical accessibility challenge for the deaf community.

EA5 (Familiarity): Many of the challenges, such as solving the specific MemoryError issues with a custom data generator and memory-mapping, were unfamiliar and required solutions beyond standard textbook examples.

5.5 Summary

This chapter has provided a comprehensive analysis of the project's adherence to professional engineering standards, its significant societal impact, and its classification as a complex engineering problem. We have justified our selection of software, hardware, and communication standards and discussed the ethical considerations and sustainability of the system. The financial analysis highlighted the cost-effectiveness of our approach. Finally, by mapping the project's challenges and activities to formal engineering criteria, we have demonstrated that this work involved in-depth analysis, innovation, and the successful application of specialist knowledge to solve a novel and complex problem.

Chapter 6

Conclusion

This chapter provides a final summary of the research project, consolidating the key achievements and contributions. It will also offer a critical reflection on the limitations of the current system and propose specific, actionable directions for future work to build upon the successes of this project.

6.1 Summary

This research project successfully demonstrates a complete development and scaling cycle for a sign language recognition system. The work was structured in two distinct phases. In the first phase, we established a new state-of-the-art performance on the public BDSLw60 v2 dataset, with our hybrid Conv1D-BiLSTM with Attention model achieving an exceptional **97.3% accuracy**, thereby validating our architectural choices.

The second, more complex research phase addressed the practical challenge of vocabulary expansion. We successfully collected a new dataset of 39 words and developed a robust, multi-stage fine-tuning methodology to integrate this new vocabulary. By systematically applying techniques like class weighting and gradual unfreezing, we overcame the significant challenges of catastrophic forgetting and data imbalance, culminating in a final model with a **93% overall accuracy** across the expanded 99-word vocabulary. The final, high-accuracy model was deployed into a robust web application, engineered to handle real-world video variability, thus transforming our research from a theoretical construct into a practical communication aid. This work provides a proven framework for building and, more importantly, scalably expanding high-accuracy sign language recognition models for real-world application.

6.2 Limitation

Despite the successful outcomes, it is important to acknowledge the limitations of the current system:

Limited Vocabulary: While expanded to 99 words, the vocabulary is still a small fraction of the full Bangla Sign Language lexicon. The system cannot interpret words outside of its training set.

Word-Level Recognition: The current model is designed to recognize isolated, word-level signs. It does not understand the grammatical structure, syntax, or context of continuous, sentence-level sign language.

Lack of Fingerspelling: The system does not have the capability to recognize fingerspelling, which is used to sign names, places, and words that do not have a dedicated sign.

Environmental Dependency: The accuracy of the feature extraction pipeline is sensitive to environmental conditions. Poor lighting, cluttered backgrounds, or obstructions (e.g., hands moving out of the camera frame) can degrade performance.

Signer Dependency: The model was trained on a specific set of signers. Performance may vary when used by new individuals with different signing styles, speeds, or regional dialects.

6.3 Future Work

The limitations of the current system provide a clear and exciting roadmap for future research and development. The following are key areas for future work:

1. Continuous Vocabulary Expansion: The fine-tuning framework we have developed can be used to continuously add new words to the model. A future project could involve building a community-sourcing platform to allow native signers to contribute new video data, organically growing the lexicon over time.
2. Transition to Continuous Sign Language Recognition (CSLR): The next major research step is to move from isolated word recognition to understanding full sentences. This would involve researching more advanced architectures, such as Transformer networks, which have shown great promise in learning the complex grammatical and contextual relationships in continuous video streams.
3. Fingerspelling Recognition Module: A separate, dedicated model could be trained to recognize the individual handshapes of the BdSL alphabet. This module could be integrated into the system, allowing it to spell out words it does not know, which would dramatically increase its practical utility.
4. Mobile Application Development: To make the tool truly accessible and portable for everyday use, the next logical step in deployment is to develop a lightweight, efficient mobile application for both Android and iOS platforms.
5. Robustness and Personalization: Future research could focus on data augmentation techniques that simulate different lighting conditions and backgrounds to make the model more robust. Additionally, a personalization feature could allow the model to fine-tune itself to a specific user's unique signing style over time.

References

[1] M. Jubayer, A. K. M. Masum, C. A. Hossain, et al., "A Deep Dive into Word-level Bangla Sign Language Recognition," *arXiv preprint arXiv:2105.05634*, 2021.

[2] P. K. Mondal, M. S. Islam, and S. M. M. Islam, "Bangla Sign Language Recognition using Convolutional Neural Network," in *2021 International Conference on Automation, Control and Mechatronics for Industry 4.0 (ACMI)*, 2021, pp. 1-6.

[3] R. Rastgoo, K. Kiani, and S. Escalera, "3D-CNN-based short-term sign language recognition from RGB-D videos," *Soft Computing*, vol. 25, no. 8, pp. 6157–6171, 2021.

[4] S. Athar, S. R. R. Yna, and F. A. K. Jilani, "Word-level sign language recognition using a deep CNN-LSTM model," *Neural Computing and Applications*, vol. 34, no. 14, pp. 11637–11651, 2022.

[5] O. Koller, J. Forster, and H. Ney, "Deep Sign: Hybrid CNN-HMM for Continuous Sign Language Recognition," in *Proceedings of the British Machine Vision Conference (BMVC)*, 2016, pp. 20.1-20.12.

[6] J. Howard and S. Ruder, "Universal Language Model Fine-tuning for Text Classification," *arXiv preprint arXiv:1801.06146*, 2018.

[7] M. De Coster, K. D'Oosterlinck, E. Van Messem, and J. Van de Walle, "Sign Language Recognition with Transformer Networks," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2021, pp. 385–393.

[8] D. Li, X. Wei, C. Wang, et al., "Word-level Deep Sign Language Recognition from Video: A New Large-scale Dataset and Methods," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2020, pp. 1449–1458.

[9] P. R. Pupale, S. D. Sapkal, and K. S. Oza, "A Review of Datasets for Indian Sign Language," in *2023 International Conference on Intelligent Systems for Communication, IoT, and Security (ICISIoTS)*, 2023, pp. 1-6.

[10] N. Adaloglou, T. Chatzis, and I. Papastratis, "A Comprehensive Study on Deep Learning-based Methods for Sign Language Recognition," *IEEE Access*, vol. 8, pp. 89286–89307, 2020.

[11] M. F. Yasir, M. A. H. Akhand, and K. Murase, "An Automated System for Recognition of Bangla Sign Language (BdSL) Alphabets and Numerals Using a Custom CNN Architecture," *Journal of Imaging*, vol. 7, no. 9, p. 177, 2021.

[12] M. H. Rakib, M. S. Ali, M. A. Islam, and M. A. H. Akhand, "A Lightweight Convolutional

Neural Network for Bangla Sign Language Alphabets Recognition," in *2022 International Conference on Advancement in Electrical and Electronic Engineering (ICAEEE)*, 2022, pp. 1-6.

[13] M. Al-Hammadi, G. Muhammad, M. A. Hossain, and A. Almogren, "Deep learning-based approach for Arabic sign language recognition," *IEEE Access*, vol. 8, pp. 88390–88401, 2020.

[14] R. Elakkiya, K. Selvamani, and S. S. S. Priyadarshini, "A survey on various methods for Indian Sign Language recognition," *Journal of Engineering Sciences*, vol. 14, no. 2, pp. 1-13, 2023.

[15] O. Sincan and H. Keles, "A survey on deep learning-based sign language recognition," in *2020 4th International Conference on Computer Science and Engineering (UBMK)*, 2020, pp. 232–237.

[16] M. R. Karim, S. M. S. Islam, and M. S. Miah, "Recognition of Bangla Sign Language Alphabets using a Deep Convolutional Neural Network," in *2022 IEEE 13th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, 2022, pp. 0588-0593.

[17] S. Tasnim, S. A. Lubna, and M. I. H. Sarkar, "A Real-Time Translator for Bangladeshi Sign Language using Convolutional Neural Network," in *2021 2nd International Conference on Robotics, Electrical and Signal Processing Techniques (ICREST)*, 2021, pp. 581-585.

213-15-4489

ORIGINALITY REPORT

12%	8%	6%	7%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Daffodil International University Student Paper	4%
2	Submitted to United International University Student Paper	<1%
3	dspace.daffodilvarsity.edu.bd:8080 Internet Source	<1%
4	Submitted to University of Finance – Marketing Student Paper	<1%
5	Pushpa Choudhary, Sambit Satpathy, Arvind Dagur, Dhirendra Kumar Shukla. "Recent Trends in Intelligent Computing and Communication", CRC Press, 2025 Publication	<1%
6	digitalarchives.aum.edu Internet Source	<1%
7	arxiv.org Internet Source	<1%
8	dokumen.pub Internet Source	<1%
9	www.mdpi.com Internet Source	<1%
10	Abdulelah Ali Alkhoraif, Mansour Alsulaiman, Wadood Abdul, Mohamed Bencherif. "Ensemble transformer-based word-level sign	<1%