Sign Language Recognition in Low-Resourced Languages

M.A. Neethamadhu

Sign Language Recognition in Low-Resourced Languages

Mr. M.A. Neethamadhu
Index No.: 20001223
Reg. No.: 2020/CS/122

Supervisor: Dr. B.H.R. Pushpananda
Co-supervisor: Dr. Ruvan Weerasinghe

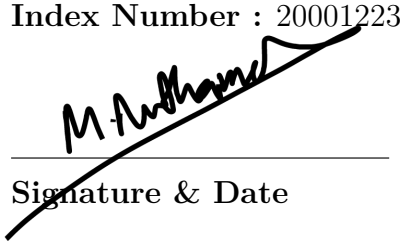
April 2025

Submitted in partial fulfillment of the requirements of the
B.Sc. (Hons) in Computer Science Final Year Project (SCS4224)

Declaration

I certify that this dissertation does not incorporate, without acknowledgement, any material previously submitted for a degree or diploma in any university and to the best of my knowledge and belief, it does not contain any material previously published or written by another person or myself except where due reference is made in the text. I also hereby give consent for my dissertation, if accepted, to be made available for photocopying and for interlibrary loans, and for the title and abstract to be made available to outside organizations.

Student Name : M.A. Neethamadhu
Registration Number : 2020/CS/122
Index Number : 20001223



Signature & Date

This is to certify that this dissertation is based on the work of M.A. Neethamadhu under my supervision. The thesis has been prepared according to the format stipulated and is of acceptable standard.

Supervisor Name : B.H.R. Pushpananda

30 - 06 - 2025

Signature & Date

Contents

List of Figures

List of Tables

Acronyms

3DCNN 3-Dimensional Convolutional Neural Networks. 7

ANN Artificial Neural Networks. 6

ArSL Arabic Sign Language. 6

ASL American Sign Language. 2, 6, 12

BSL British Sign Language. 2

CSLR Continuous Sign Language Recognition. 6, 49

DL Deep Learning. 3, 11

FSL Flemish Sign Language. 11

HMM Hidden Markov Model. 6

IrSL Irish Sign Language. 6

ISL Indian Sign Language. 1, 12, 15, 20, 26, 30–33, 37, 38, 40, 41, 43, 46, 47

ISLR Isolated Sign Language Recognition. 6, 7, 49

LSTM Long Short Term Memory. 7, 17

NLU Natural Language Understanding. 7

PoS Parts of Speech. 6

RVM Relevance Vector Machines. 6

SLR Sign Language Recognition. 1–11, 42, 44, 46–49

SSL Sinhala Sign Language. iv, 1–4, 8–11, 13, 15, 17, 19, 20, 22, 23, 26, 30–34, 37, 38, 40–43, 46–49

SVM Support Vector Machine. 6

1 Abstract

Sign Language Recognition (SLR) systems are vital for bridging communication gaps between deaf and hearing communities, yet low-resourced languages like Sinhala Sign Language (SSL) face challenges due to limited training data. This thesis investigates the efficacy of cross-lingual transfer learning to enhance SLR accuracy for SSL, a language with scarce datasets. By pre-training Transformer-based models on a large Indian Sign Language (ISL) dataset and fine-tuning them on a 64-word SSL dataset, the study evaluates performance improvements in low-resource scenarios, simulating data scarcity with 2 to 6 instances per class. Results demonstrate that transfer learning significantly boosts accuracy with 2 or 3 instances per class, achieving up to an 8% improvement over models trained directly on SSL, though benefits diminish with 4 or 6 instances per class. The study also explores the impact of overlapping semantic and movement patterns between ISL and SSL, finding no conclusive advantage. Additionally, varying base model sizes (80 to 240 classes) showed no consistent effect on fine-tuning performance, suggesting further research is needed. This work contributes to the field by providing insights into transfer learning strategies for low-resourced SLR, offering methodologies applicable to other under-resourced sign languages, and highlighting the potential for developing accessible communication systems with minimal data. In conclusion, this study confirms that pre-training on high-resource sign languages like ISL can lead to meaningful improvements in recognizing signs from SSL, particularly in extreme low-resource conditions where only 2–3 video instances per sign are available. While the benefits of transfer learning diminish as more training data becomes available, this approach offers a promising pathway for developing effective SLR systems for underrepresented languages with limited datasets.

2 Introduction

Over 400 million people worldwide experienced hearing loss, and to bridge the communication gap, they utilized over 300 distinct sign languages. These languages enabled the deaf community to communicate with each other and with hearing individuals. However, despite the significant number of sign language users, a substantial communication gap persisted between the hearing and deaf communities. This was primarily due to the limited number of individuals proficient in sign language in daily interactions. This underscored the importance of Sign Language Recognition (SLR) systems. The development of such systems was critical as they facilitated bridging this communication gap. Over the years, extensive research had been conducted in this field, leveraging advancements in computer science, including machine learning and deep learning.

Most existing SLR solutions were developed for sign languages with abundant training data, such as American Sign Language (ASL) and British Sign Language (BSL). However, with over 300 sign languages worldwide, many were used by smaller communities and lacked extensive training datasets. Sinhala Sign Language (SSL) was one such low-resourced language. Developing SLR systems for these languages posed challenges due to their unique grammatical structures and distinct signs for the same concepts compared to other sign languages. For instance, the sign for "evening" in SSL differed from that in ASL. Consequently, universal solutions based on data from high-resourced languages like ASL or BSL were often ineffective for low-resourced languages.

2.1 Motivation

In Sri Lanka, over 300,000 individuals with hearing disabilities primarily used SSL to communicate with each other and with hearing individuals. However, only a small fraction of hearing individuals were proficient in this sign language. As a result, communication difficulties between these groups were common in Sri Lanka. For example, observations at certain cafeterias in Sri Lanka (Figure 2.1), staffed by employees with hearing loss, revealed that, despite signboards displaying commonly used signs and their translations, effective communication remained challenging. A reliable sign language translation system could have facilitated communication between these groups in society. However, the primary obstacle to developing such systems was the scarcity of data in SSL. Consequently, this study was undertaken to explore methods for training SLR systems with minimal data.

Figure 2.1: Pizza Hut embraces inclusivity. Ref: *ft.lk* (2019)

2.2 Problem Definition

Training a Deep Learning (DL) model to classify or recognize sign language typically required large datasets, often comprising more than 15 [1] instances per class on average. For a practical vocabulary, datasets needed to include at least 50 to 100 words. Constructing such datasets presented significant challenges, particularly in reducing bias, ensuring generalizability, and managing background noise. Therefore, it was valuable to investigate technical approaches for training models with small datasets, especially for low-resourced languages, where datasets often contained extremely low number of instances per word (class).

2.3 Aims and Objectives

The primary aim of this study was to investigate the potential of cross-lingual transfer learning to enhance the accuracy of SLR systems for SSL. Specifically, this study sought to:

1. Evaluate the effectiveness of cross-lingual transfer learning for Sinhala Sign Language (SSL): A base model was trained on a high-resourced sign language and fine-tuned for SSL to improve SLR accuracy.

2. Contribute to the broader field of low-resourced SLR: The study provided insights and methodologies applicable to other low-resourced sign languages, thereby advancing research in this domain.

[1] This number is based on datasets considered during this study

This study synchronized SSL with global advancements in SLR technology for low-resourced sign languages and aimed to surpass the performance of existing methods. The research made significant contributions to the field of SLR and benefited the large community of low-resourced sign language users.

3 Literature Review

3.1 Literature Review

3.1.1 Overall structure of SLR domain

SLR is divided into various branches that play important roles in understanding and interpreting sign language. Each of these sub-branches has its own implementation methodology and theoretical concepts. According to Elakkiya (2021), there are four major branches in this topic. However, since most modern methodologies like deep learning rely on datasets, preparing proper datasets can also be considered as an area in this field. Overall the field looks like this as shown in 3.2.

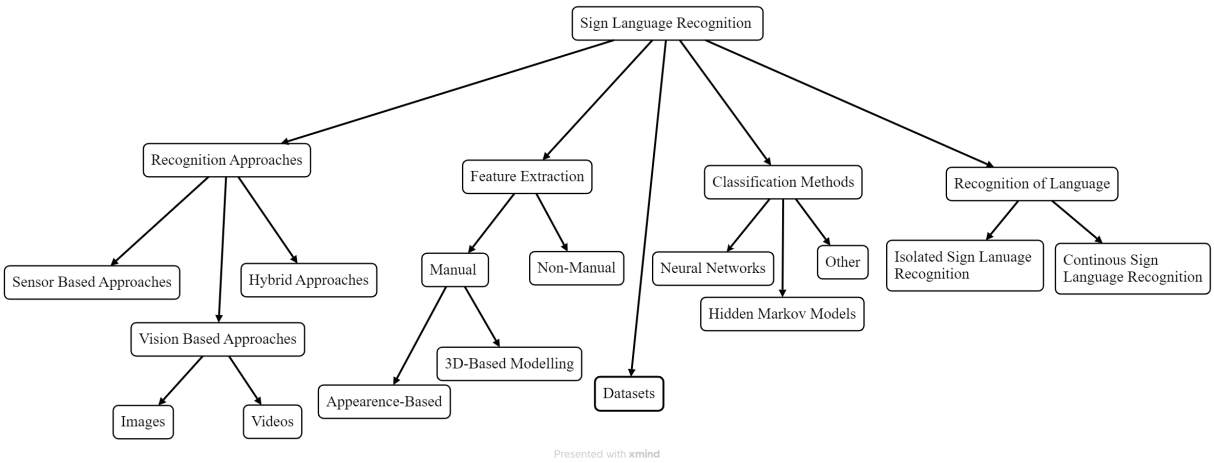


Figure 3.2: Sign Language Recognition Branches

Recognition approaches in SLR include various methods to recognize and interpret sign language gestures, classified into sensor-based, vision-based, and hybrid approaches. Sensor-based techniques use sensors like gloves used in Quesada et al. (2017) and Madushanka et al. (2016) , while vision-based approaches rely on computer vision techniques as in Hu et al. (2023) and Kumar & Bajpai (2023). Vision based methods are further divided into static images and dynamic videos. Hybrid approaches combine both technologies for better accuracy. Each approach has its own pros and cons.

Feature Extraction in SLR involves capturing attributes from hand gestures and facial expressions, making them manageable for analysis. Features include hand shape, motion trajectories, spatial details, and facial expressions. This can be divided into Manual (appearance-based and 3D-based models) and Non-Manual feature extraction. Studies such as Elmezain et al. (2008), Haputhanthri et al.

(2022) and Coogan & Sutherland (2006) have investigated the deep insights between these feature extraction methods.

The classification component identifies the actual sign post-feature extraction. According to Elakkiya (2021), methods include Artificial Neural Networks (ANN) (Elmezain et al. (2008), Cicirelli & D'Orazio (2017)), Hidden Markov Model (HMM) (Cicirelli & D'Orazio (2017), Tur & Keles (2021)), Support Vector Machine (SVM) (Dardas & Georganas (2011)) , and Relevance Vector Machines (RVM) (Wong & Cipolla (2005)). This process is crucial for mapping features to signs.

Recognition is divided into Isolated Sign Language Recognition (ISLR) and Continuous Sign Language Recognition (CSLR). ISLR translates signs word by word, using static images or videos. This area has been stuided in most researches such as Aly & Aly (2020). CSLR recognizes full sentences at once, which is more useful but complex. Since this is the most desirable version of SLR studies such as Pramanto & Suharjito (2023) have tried to comeup with solutions for this problem.

Datasets are vital for SLR research. For example, Li et al. (2020) created a large American Sign Language (ASL) dataset for word-level recognition, and Latif et al. (2019) developed a labeled Arabic Sign Language (ArSL) image dataset. Liyanaarachchi et al. (2021) also attempted to create a dataset for Sign Language Recognition (SLR).

In addition to these main branches, there are subfields such as SLR for low-resourced languages. Most of the sign languages in the world are used by smaller communities. Therefore, it is inevitable to have smaller or no datasets for these sign languages although developing SLR systems for these languages is a requirement. Therefore, a few researchers have been conducted to study methodologies to develop SLR systems for these low-resourced languages and so far there has been improvement in this area as there is a requirement for such low resourced systems.

3.1.2 Low resourced sign language recognition

Holmes et al. (2023) conducted a study regarding how transfer learning would create linguistic relationships between American Sign Language (ASL) and Irish Sign Language (IrSL) which is a low-resourced language. Here, they have employed a 5-staged model that is made with 1D CNN, keypoint embeddings, and transformers. Their study shows the usefulness of primary datasets containing or articulated signs. Furthermore, the most important finding of their research is the relationship between the frequency of overlapping glosses/lemmas/Parts of Speech (PoS) tags between datasets. It seemed that if the distribution is closer to each other, the accuracy of the fine-tuning shows an improvement. This is an

important observation that helps to select a primary dataset when doing research in transfer learning.

Another study conducted by Selvaraj et al. (2021) shows how techniques used in Natural Language Understanding (NLU) can be applied for SLR. Pretrained encoders such as BART are used in NLU. Similarly Selvaraj et al. (2021) proposes a standardized pose-extractor to be used as an encoder. They trained this encoder using a large corpus of sign language data in Indian Sign Language under self-supervised methods. Then this model is fine-tuned using other sign languages such as Argentinian Sign language. Here, authors have shown the improvements between 3% to 18% based on the fine-tuned language.

Coster et al. (2023) has done research on a robust method for extracting sign embeddings from low-resource languages. Here, they have addressed the issues when using pose estimation models such as OpenPose by introducing solutions for dealing with missing points. Then they used a ResNet and transformers-based architecture to create pose embeddings and detect ISLR. They have conducted to experiments for both fixed and variable length sequences and shown this method outperforms image-based models that were previously more powerful than keypoint-based models for SLR.

Another interesting study is conducted on identifying new signs of inferring time using Zero-shot learning. effectiveness of this method is described in Bilge et al. (2019). Here, they have used textual descriptors of sign classes in order to realize seen-to-unseen class transfer. They have used a model with two main components. One is to learn visual data using 3-Dimensional Convolutional Neural Networks (3DCNN) and Long Short Term Memory (LSTM). The other component is to learn the embedding of the visual reprxesentation to the closest text description. However, this method does not show any improvement compared to other methods since it is difficult to differentiate signs using textual descriptions because hand movement differences might be subtle.

3.1.3 Other related recent studies

A study conducted by Moryossef et al. (2021) evaluates the applicability of pose estimations for SLR. This study has used multiple pose estimation models and multiple classification networks. All the results show over 80% accuracy in SLR tasks, hence this study is a good example of proving the power of pose estimation models. Also, this study cites a library that contains tools for process pose data.

However, the study conducted by Coster et al. (2023) finds a few ways to improve the output of pose estimation models using methods like imputation and normalization. There, they show the significance of the post-processing of the skeleton points before using them for SLR tasks.

3.2 Research Gap

Most studies on SSL required large datasets to achieve optimal performance. However, SSL, as a low-resourced language, lacked such datasets, resulting in suboptimal performance in existing SLR systems. The primary barriers to creating large datasets for SSL included limited data sources and privacy concerns. Consequently, there was a significant gap in the application of low-resource SLR methods to SSL. This study explored methods such as cross-lingual transfer learning, and to a lesser extent contrastive learning, to elevate the accuracy of SLR systems for SSL to competitive levels.

This research investigated the potential of cross-lingual transfer learning for SLR in SSL and assessed its improvements over existing SSL studies.

4 Research Questions and Contribution

4.1 Research Questions

1. **Is it possible to improve the accuracy of Sinhala Sign Language (SSL) models using cross-lingual transfer learning?**

   Cross-lingual transfer learning had shown improvements in other sign languages, leading this study to investigate its applicability to SSL. The research evaluated the extent of accuracy improvements, established baseline results using small datasets, and analyzed the effectiveness of this approach for SSL.

2. **What underlying patterns and trends can be uncovered through transfer learning techniques?**

   This study intended to examine the most effective transfer learning techniques (such as what parameters are the best, any trends with different parameters) for SLR and quantified the accuracy improvements achieved, providing insights into their applicability for low-resourced sign languages like SSL.

3. **What are the most suitable primary languages for pre-training a model to recognize Sinhala Sign Language (SSL)?**

   Different sign languages have different gestures and signs for the same word. Different primary languages may share varying degrees of similarity in signs or sign components with low-resource sign languages. This question aimed to identify which primary language is most suitable for SSL so that transfer learning methods can be effectively implemented with that language. [2]

4.2 Significance of the Project and Research Contribution

4.2.1 Research Contribution to Computer Science

1. **Advancement in Transfer Learning**

   This study explored cross-lingual transfer learning inSLR, contributing valuable insights on optimizing these strategies for low-resourced sign languages like SSL. Identifying suitable primary languages for model pre-training and fine-tuning will inform best practices in this area.

[2]However, due to tim constraints it was possible to explore this only on single primary language

2. **What are the most effective transfer learning methods for SLR and how much it is possible to improve**

   By adapting feature extraction and fine-tuning, this study tried to show the effectiveness of transferring knowledge between different sign languages.

4.2.2 Research Contribution to Society

1. **SSL Recognition Systems with Minimal Data**

   By following the methodologies explored in this project, it may be possible to develop effective SLR communication systems for SSL with minimal training data. This is crucial for low-resourced languages and represents a significant advancement in creating accessible SLR technologies for small language communities.

2. **Educational Empowerment**

   Using enhanced SLR systems it is possible to develop learning platforms for SSL as well as normal people to learn SSL. This can further cause to connect people who are having hearing loss with rest of the population.

5 Methodology and Data Analysis

5.1 Research Approach

The research approach adopted in this project was based on an Experimental Research Methodology, which focused on the empirical evaluation of Deep Learning (DL) techniques to assess their effectiveness in recognizing SSL. Specifically, this study investigated the impact of cross-lingual transfer learning by comparing the performance of models trained with and without transfer learning strategies.

The goal was to determine whether cross-lingual transfer learning could achieve measurable improvements in accuracy for recognizing SSL. This approach involved a structured process of defining the problem, selecting appropriate datasets and models, applying transfer learning techniques, and evaluating the outcomes using quantitative measures, primarily accuracy.

Unlike Design Science Research, which emphasizes the creation of novel artifacts, this study did not propose new model architectures or systems. Instead, it contributed to the body of knowledge by providing experimental evidence and practical insights into the applicability and benefits of cross-lingual transfer learning in the context of SSL.

The high-level plan of the research approach was executed as follows. Large sign language datasets were identified to train base models. The selected datasets were used to train multiple SLR models. Baseline models were then trained using an SSL dataset. Subsequently, the base models were fine-tuned with the same SSL dataset, and the results were collected. Finally, these results were analyzed to identify meaningful relationships. The flow of this process is illustrated in Figure 5.1.

Initially, the study planned to evaluate different model architectures and their performance in cross-lingual transfer learning for SLR, including hyperparameter tuning, to explore optimal configurations. Also trying out multiple different primary languages and compare different languages was also a target. However, this objectives were excluded due to time constraints.

5.2 Methodology

5.2.1 Datasets

The first dataset evaluated was VGT (2022), which contained signs in Flemish Sign Language (FSL). This dataset included longer video sequences captured from various angles with multiple signers in the same video, as shown in Figure 5.2. Despite its large size, the dataset's raw nature required extensive preprocessing, including manual video inspection. Challenges included the presence of multiple

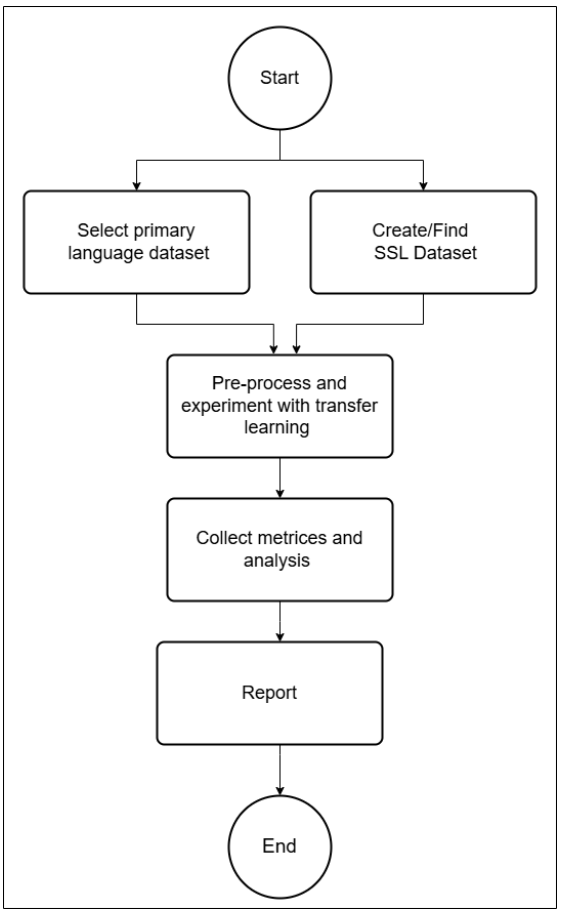


Figure 5.1: High-level research approach

individuals in videos, inconsistent skeleton point extraction due to varying camera angles, and difficulties in extracting individual words for an ISL dataset due to complex annotations. Consequently, this dataset was deemed unsuitable for the study.

The second dataset evaluated was Microsoft (2018), an ASL dataset created by Microsoft, containing metadata for sign videos sourced from the internet. Acquiring this dataset required writing a script to collect videos from URLs, and several GitHub repositories provided solutions for this task. However, most video URLs were expired, rendering the results unsatisfactory. Therefore, this dataset was not used.

The third dataset evaluated was *WLASL (World Level American Sign Language) Video* (2020), which also comprised videos from various sources and included 2000 isolated signs in ASL. Unlike the Microsoft (2018) dataset, a considerable amount of data was acquired using an initial script. However, similar issues with video accessibility persisted, and after conducting preliminary experiments, this dataset was also excluded.

The fourth dataset selected was Sridhar et al. (2020), an ISL dataset for Indian

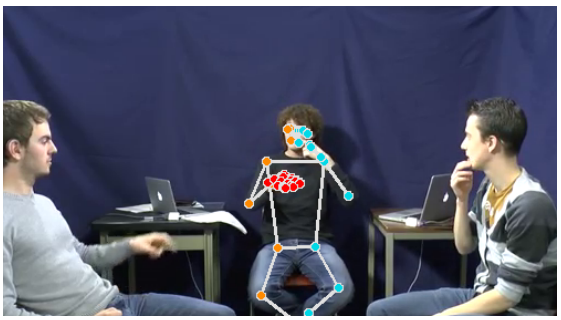


Figure 5.2: Skeleton extraction was challenging with multiple people in videos

Sign Language (ISL), containing 263 classes categorized under 15 sections, such as adjectives, colors, and people. This dataset was collected using actual Indian signers, with videos properly edited to include only the signing portion from a frontal view. Additionally, the dataset was directly downloadable, avoiding issues like expired URLs. The signs corresponded to the English meanings of the Indian words, making it suitable for cross-lingual transfer learning. Consequently, this dataset was chosen as the primary large dataset for the study.

For SSL, representing a low-resourced language, several datasets provided by UCSC were available, including both isolated signs and sentence-level signs. From these, a 64-word video dataset, originally constructed in a previous study (Alwis (2023)), was selected as the primary SSL dataset for model training. However, imperfections were identified, leading to the removal of the word ඔව් (meaning "yes" in Sinhala) from the dataset. Additionally, the meaning of the word "w064" could not be determined but later identified as no sign video as a fallback option. Although the study aimed to train models with minimal data, a sufficient number of samples was still required for effective model validation. A summary of this dataset is provided in Table 5.1.

Code	**Sinhala Word**	**English Meaning**
w001	ඔයාට	To_You
w002	ඔයා	You
w003	මම	Me
w004	අම්මා	Mother
w005	ඔබේ	Your
w007	එය	It
w008	නංගී	Sister
w009	විශ්වවිද්‍යාලය	University
w010	ලොකුයි	Big
w011	හොඳයි	Good
w012	ටිකක්	Little

Code	Sinhala Word	English Meaning
w013	අද	Today
w014	දැන්	Now
w015	හෙට	Tomorrow
w016	පාසැල	School
w017	ලමය	Children
w018	සිංහල	Sinhala
w019	පංති	Class
w020	දහදෙනෙක්	Ten_People
w021	වාරතාවක්	Report
w022	වීඩියෝව	Video
w023	සම්බන්දතාව	Relationship
w024	සන්වාදකොටුව	Dialog box
w025	ඉංග්‍රීසි	English
w026	විභාගය	Exam
w027	අඟහරුවාදා	Tuesday
w028	වයස	Age
w029	ගෙදර	House
w030	මගේ	My
w031	නැවත	Again
w032	ඉන්නේ	Staying
w033	උගන්වනවා	Teach
w034	ඉගෙනගන්නවා	Learn
w035	කතාකරනවා	Talk
w036	යනවා	Go
w037	කරන්න	Do
w038	හදනවා	Make
w039	ආයුබෝවන්	Hello
w040	පේනවා	See
w041	සමාවෙන්න	Sorry
w042	නැහැ	No
w043	කියන්න	Tell
w044	ලියන්න	Write
w045	පටන්ගන්නවා	Start
w046	කියවන්න	Read
w047	ස්තුතියි	Thank_You
w048	කොහොමද	How
w049	කොහේද	Where
w050	මොකද්ද	What
w051	කියක්ද	How_many

Code	Sinhala Word	English Meaning
w052	ඉගෙනගන්නවද	Learn?
w053	පුලුවන්ද	Can?
w054	ඉන්නවද	Stay?
w055	පේනවද	See?
w056	කවදද	When
w057	කතාකරනවාද	Talk?
w058	විස්ස	Twenty
w059	තියෙනවා	There
w060	කරුණාකර	Please
w061	තේරුනේ	Understand
w062	නිවාඩු	Holiday
w063	ඔව්	Yes
w064	-	Still

Table 5.1: SSL dataset and english meanings

5.2.2 Data Preprocessing

Data cleaning and preprocessing were critical steps in this study. For both the INCLUDE dataset (ISL) and the SSL dataset, similar preprocessing techniques were applied, as both datasets contained comparable video formats. The steps were as follows:

Google's MediaPipe library *MediaPipe Solutions guide* (2024) was used to extract landmarks from the video files. This library, developed over several years, provided reliable output for the datasets. Specifically, the MediaPipe holistic model was employed, generating 543 landmarks for pose, face, and hands. These included 33 points for the full-body pose (covering the torso, arms, and legs), 21 points each for the left and right hands, and the remaining points for the face mesh. The output consisted of (x, y, z) coordinates. However, the z-axis values were discarded due to the absence of depth data in the video files. The x and y coordinates were used to represent the points extracted by the holistic model, as illustrated in Figure 5.3.

Although skeleton-drawn videos could have been used for model training, only the (x, y) coordinates were utilized for all downstream tasks due to several advantages. First, this approach mitigated biases from background variations, the signer's skin tone, and facial features, effectively eliminating person-dependent features. Second, it required significantly less computational power, which was beneficial given the numerous training experiments conducted. However, this method resulted in a slight reduction in model accuracy, a trade-off accepted based on

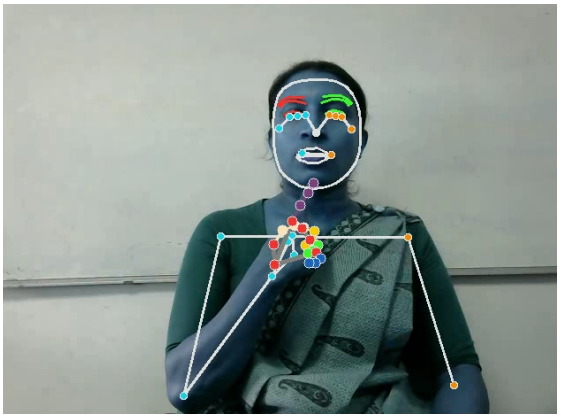


Figure 5.3: Skeleton extraction from videos

studies like Sridhar et al. (2020), which reported a 4–6% accuracy decrease for larger datasets.

Despite the frontal view of signers in the videos, slight variations in positioning along the x and y axes meant that signers were not always centered. To address this, the coordinate points were standardized to reposition the signer's skeleton to the center of the frame, regardless of their original position. This process is shown in Figure 5.4. The hand and pose skeletons were centered, but the face mesh remained in its original position, as it was not used for sign recognition. This decision simplified the model input by reducing the number of features.

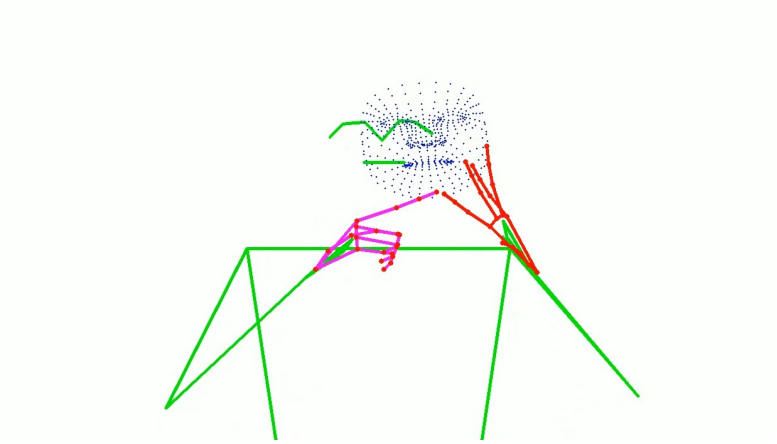


Figure 5.4: Skeleton point standardization

Once extracted, the skeletons were saved in a Python pickle file for future use. This was a one-time process and the most CPU-intensive task in the study.

5.2.3 Preliminary Experiments

Initial experiments were conducted to gain a better understanding of the datasets. After extracting all key points, 33 key points were selected to create an intermedi-

ate representation. In the early stages, these points were chosen based on intuitive understand, that's we selected points that looked most active. Those point indices are 0, 1, 3, 4, 6, 8, 10, 12, 14, 16, 18, and 20 from each hand, and indices 0, 11, 12, 13, 14, 15, 16, 23, and 24 from the pose model (body skeleton). Figure 5.5 illustrates the selected key points.

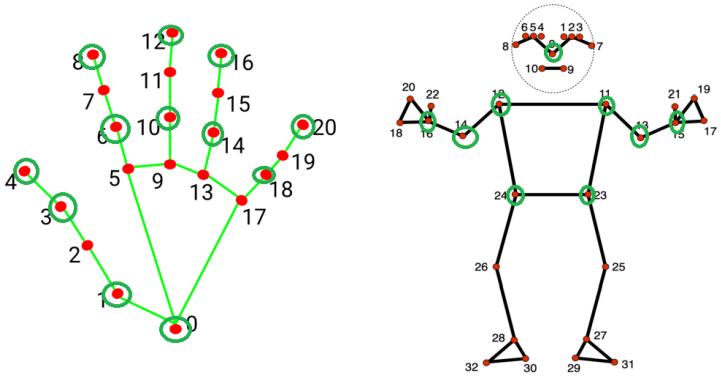


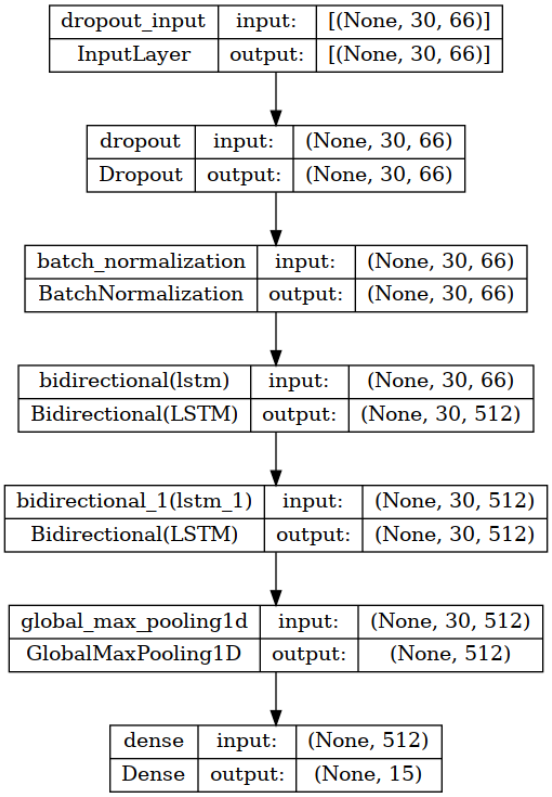
Figure 5.5: Selected skeleton points, marked in green

During the initial stages, the *WLASL (World Level American Sign Language) Video* (2020) dataset was used for preliminary experiments. A simple classification model was trained on a portion of the WLASL dataset. A script identified classes overlapping with the SSL dataset, and 20 classes (15 overlapping and 5 random) with the highest number of video samples were selected, resulting in a dataset of approximately 137 video samples.

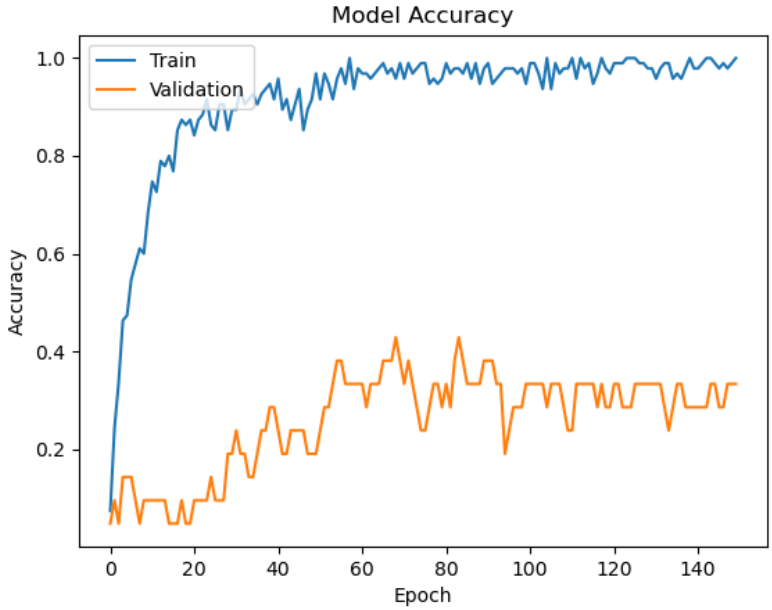
The model trained was a sequence classification network using an LSTM, as shown in Figure 5.6a. Despite testing multiple configurations, the model's accuracy remained low, primarily due to the limited dataset size, which was insufficient for effective training of deep neural networks. The resulting accuracy is shown in Figure 5.6.

The next preliminary experiment involved selecting a subset of the SSL dataset and training the same classification model exclusively on this subset. The subset comprised approximately 450 video samples across 15 classes, including a "No Sign" class with no signing gestures. These 15 classes included some words from the main SSL dataset plus a few additional words, as access to the full SSL dataset was initially limited due to technical issues. These issues were later resolved, enabling access to the complete SSL dataset with 64 classes. In this experiment, reasonably high accuracy was achieved without applying transfer learning, relying solely on the SSL dataset. The accuracy metrics and confusion matrix are presented in Figure 5.7.

These two preliminary experiments provided valuable insights into the strengths and limitations of the datasets, informing the design of subsequent experiments.

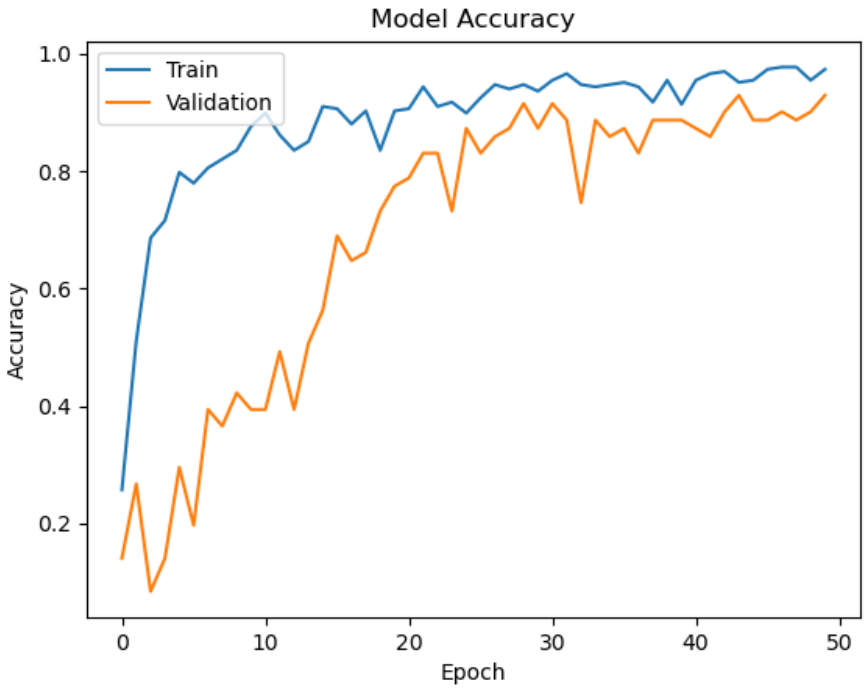


(a) Model architecture

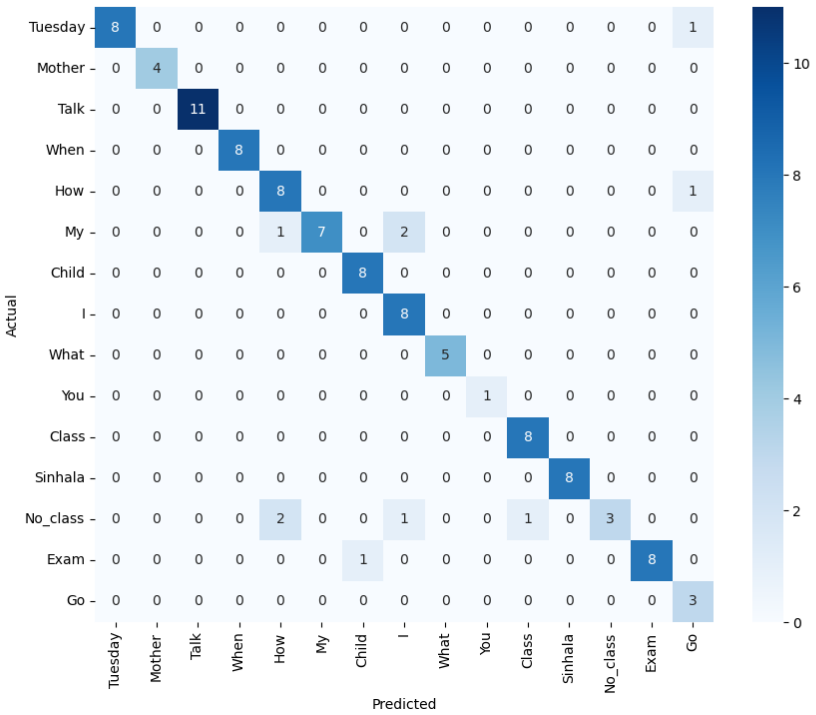


(b) Training and validation accuracies

Figure 5.6: Sequence classification network used for initial experiments.



(a) Training and validation accuracies



(b) Confusion matrix

Figure 5.7: Classification model on SSL dataset

5.2.4 Data Analysis

Comprehensive data analysis was conducted on the selected 64-class SSL dataset and the ISL dataset. This analysis enhanced understanding of the datasets and informed the design of subsequent experiments.

Overlapping Glosses Between Datasets by Meaning

To initiate the analysis, similarities between the datasets were investigated. As noted by Holmes et al. (2023), transfer learning is more effective when source and target datasets share similar vocabulary. Based on this insight, the study examined the extent of common words between the ISL (INCLUDE) and SSL datasets.

A challenge arose due to differences in word semantics. The SSL dataset originally contained words in Sinhala, which were translated into English to simplify processing. However, this introduced ambiguity, as the ISL dataset used slightly different English wordings for similar meanings. Consequently, simple string matching was insufficient.

To address this, a semantic similarity check was applied between the translated Sinhala words and the English words in the ISL dataset. Since the INCLUDE dataset contained English meanings, only the Sinhala words required translation. By setting a similarity threshold of 0.80, 20 out of the 64 words in the SSL dataset were identified as having semantically similar counterparts in the ISL dataset. These results are shown in Table 5.2 and these words were designated as **Overlapping Words** because they are in the intersection of both datasets. Figure 5.8 shows this concept visually. Additionally, to understand the distribution of the ISL dataset, a plot was created, as shown in Figure 5.9. It indicated that approximately 100 classes contained more than 15 instances, while a portion of classes had fewer than 10 instances per class among the top 240 classes.

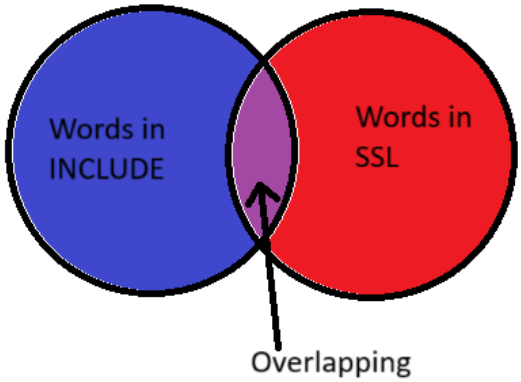


Figure 5.8: Overlapping words between two datasets

SSL Word	INCLUDE Word	Instance Count	Similarity Score
To_You	you	21	0.924
To_You	How_are_you	21	0.857
You	you	21	1.000
Mother	Mother	20	1.000
It	it	21	1.000
Sister	Sister	20	1.000
University	University	21	1.000
Big	big_large	21	0.909
Good	good	21	1.000
Little	small_little	22	0.919
Today	Today	14	1.000
Tomorrow	Tomorrow	14	1.000
School	School	20	1.000
Children	Child	20	0.833
Tuesday	Tuesday	14	1.000
House	House	21	1.000
Hello	Hello	21	1.000
Thank_You	Thank_you	21	1.000
How	How_are_you	21	0.855
What	How_are_you	21	0.827
How_many	How_are_you	21	0.872

Table 5.2: Semantically similar words between SSL and INCLUDE datasets. Instance count refers number of instance in INCLUDE dataset

Most Important Points

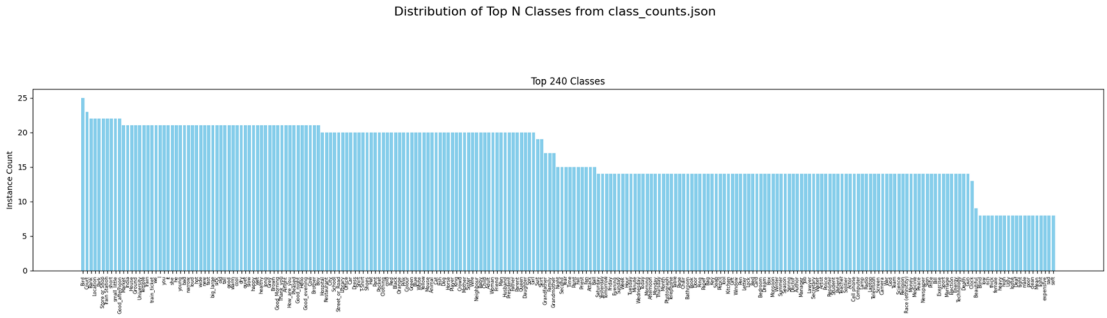


Figure 5.9: 240 words and their frequency distribution. HQ Image

As previously noted, skeleton points were initially selected for preliminary experiments based on the intuitive understanding of the researchers. At this stage, it was necessary to identify the specific skeleton points based on proper analysis.

To determine which skeleton points exhibited the greatest overall movement in sign language gestures within the SSL dataset, a comprehensive movement analysis of skeletal landmarks extracted from video data was conducted. This analysis quantified the motion of key body parts—specifically the full-body pose, left hand, and right hand—across video frames, providing insights into gesture characteristics and informing the design of subsequent experiments.

The analysis focused on computing the movement of landmarks between consecutive frames to quantify gesture dynamics. For each video, the Euclidean distance traveled by each landmark across all frame pairs was calculated, and these distances were accumulated to represent total movement. To account for variations in video length, the total movement was normalized by the number of frame intervals (i.e., the number of frames minus one), yielding an average movement per frame interval for each landmark.

For the full-body pose, movement was calculated as the displacement of each landmark between frames. For the hands, a normalization step was introduced to isolate finger and knuckle movements relative to the wrist. Specifically, the wrist's displacement was subtracted from each hand landmark's movement to focus on local hand gestures, which are critical in sign language. This approach ensured that global arm movements did not overshadow the finer articulations of the fingers.

The analysis was applied to all videos within each class (representing a specific sign or word) in the SSL dataset. For each class, the average movement of pose, left hand, and right hand landmarks was computed across all videos, providing a class-level summary of gesture dynamics. These averages were also aggregated across all classes to obtain an overall movement profile for the dataset, highlighting the most active landmarks across all signs.

To interpret the movement data, the results were visualized in a structured format. For each class and the overall dataset, plots were generated, comprising:

- *Bar Charts:* These depicted the average movement per frame interval for each landmark (33 for pose, 21 for each hand), using a unified scale to facilitate comparison across body parts. Figure 5.10 shows this for the word "English" in the SSL dataset, and Figure 5.12 presents this for the overall dataset.

- *Skeletal Representations:* These illustrated the spatial arrangement of landmarks in a selected frame, with points colored according to their movement magnitude. For the pose, the first frame was used for consistency, while for the hands, frames containing valid hand data were prioritized to capture meaningful gesture configurations. This visualization is shown in Figure 5.11 for the word "English" in the SSL dataset, and Figure 5.13 presents this for the overall dataset.

The skeletal visualizations employed a heatmap color scheme to highlight areas of high movement, with connections drawn between landmarks (e.g., between joints in the pose or fingers in the hands) to illustrate the skeletal structure. This dual representation—quantitative bar charts and qualitative skeletal plots—provided a comprehensive view of gesture dynamics, identifying the most active body parts during signing.

Purpose and Insights

The movement analysis helped quantify the relative importance of different body parts in sign language gestures, identifying landmarks with the highest motion (e.g., specific fingers or joints). Based on this analysis, the 10 most important key point indices were identified: (18, 20, 22, 16, 19, 17, 21, 15, 30, 32) for the pose, (8, 12, 16, 7, 11, 20, 15, 4, 19, 10) for the left hand, and (8, 12, 16, 7, 11, 20, 4, 15, 19, 6) for the right hand. These are shown in Figure 5.14. Unexpectedly, this indicated that two points in the right foot were among the most important. This was not anticipated, prompting multiple reviews of the code and methodology, which consistently produced the same results. Further investigation revealed that this could have occurred due to occlusion handling in the MediaPipe pose model Bazarevsky et al. (2020). This issue could have been mitigated by excluding leg and foot landmarks earlier, but at this stage, these landmarks were excluded from further consideration. Consequently, the following points were selected for continued use: (0, 15, 16, 17, 18, 19, 20) for the pose and (0, 4, 7, 8, 11, 12, 15, 16, 19, 20) for both the left and right hands. Point 0 was specifically included to capture the relative motion of the palm and body, enhancing the model's understanding of gesture dynamics.

To further validate this analysis, two transformer models (discussed in later sections) were trained: one using all 61 points and another using the selected 27

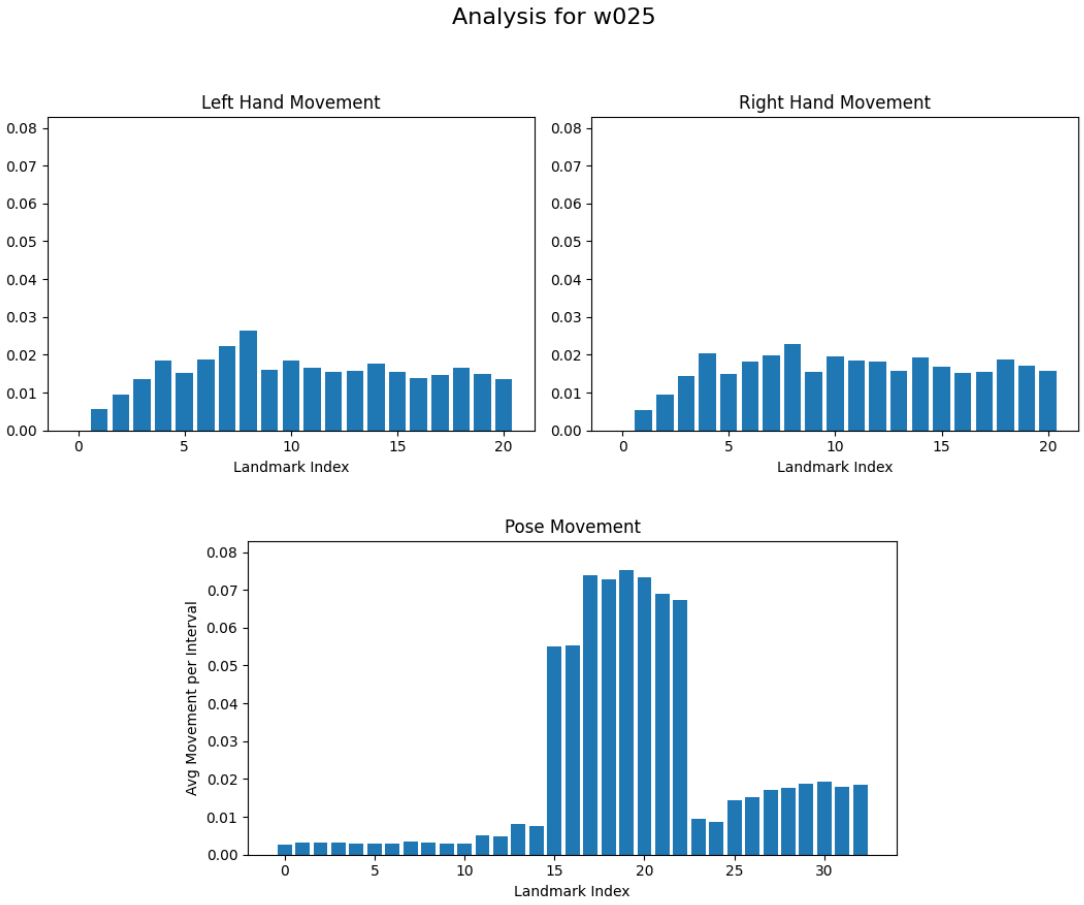


Figure 5.10: Skeleton movement bar chart analysis for w025 - English

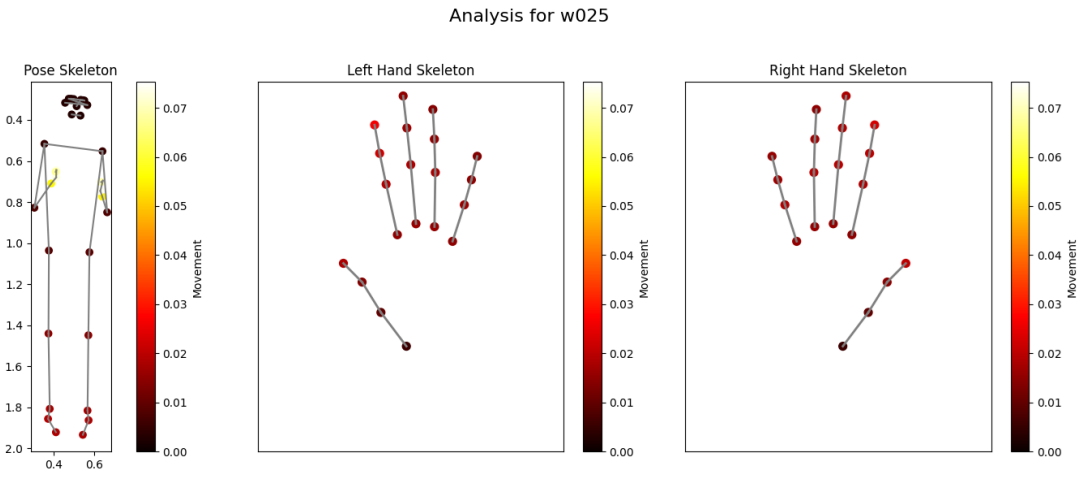


Figure 5.11: Skeleton movement skeleton representations for w025 - English

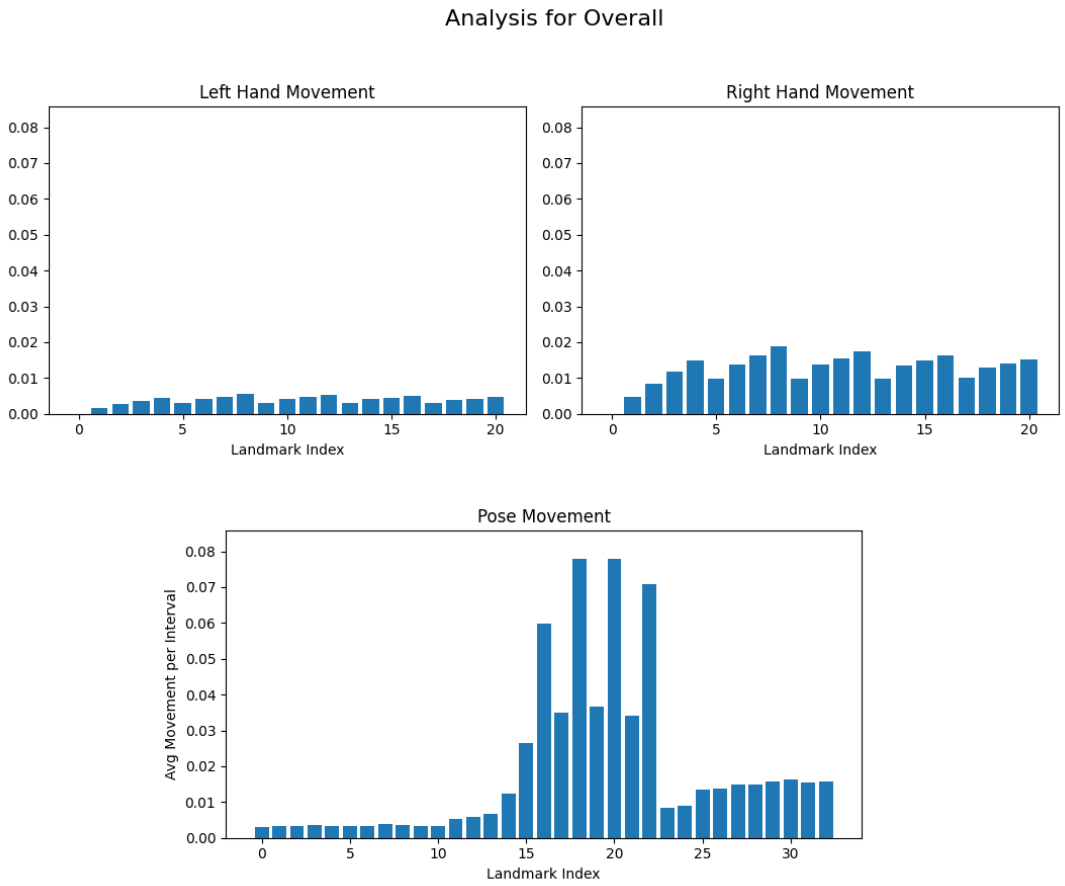


Figure 5.12: Overall skeleton movement bar chart analysis for all words

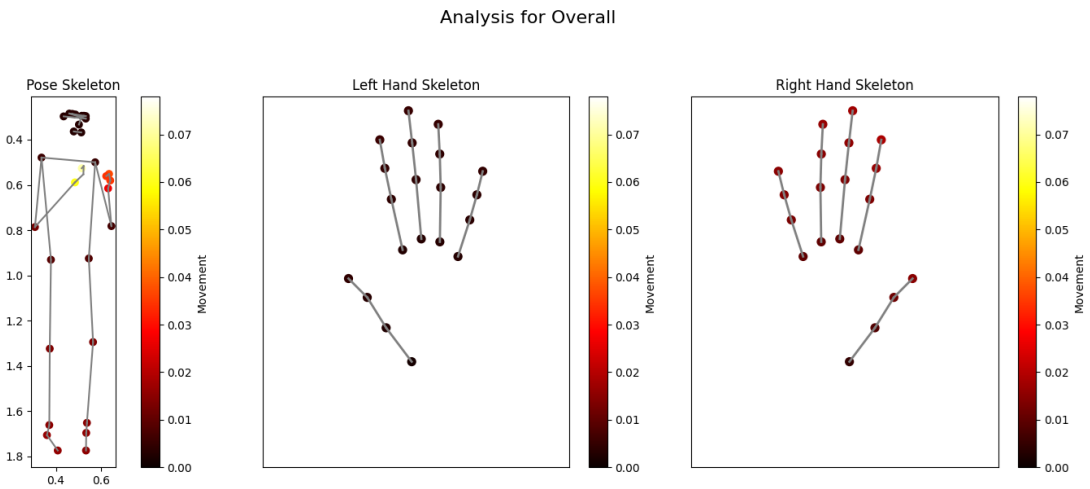


Figure 5.13: Overall skeleton movement skeleton representations for all words

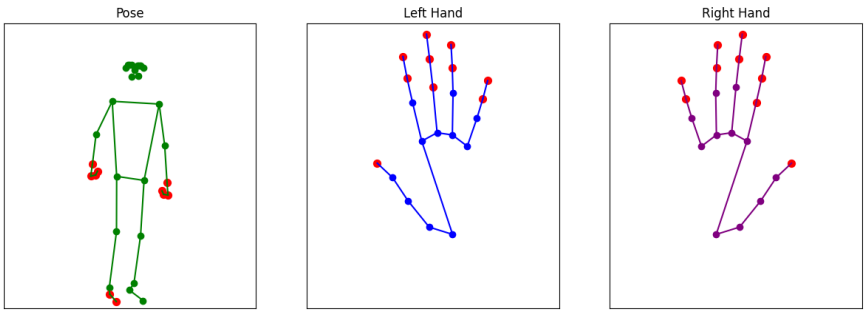


Figure 5.14: Points identified as most important

points. This experiment was conducted on the 64-class SSL dataset, with each class having varying numbers of instances to strengthen the study. The results are presented in Table 5.3 and visualized in Figure 5.15. Notably, the selected 27 points did not yield significant improvements in model accuracy. However, these points were retained for subsequent experiments, as they reduced training complexity without compromising model performance.

Number of Points	Instances	Accuracy	F1 Score
61	7	0.5385	0.5283
27	7	0.5425	0.5283
61	12	0.7385	0.7312
27	21	0.7126	0.7047
61	17	0.7693	0.7672
27	17	0.7591	0.7531

Table 5.3: Performance results across different configurations of skeleton points and instance counts. Accuracy and F1 Score values show model performance under each configuration.

5.2.5 Pre-training using INCLUDE Dataset

Once the analysis phase was completed, the next step was to train base models using the primary dataset, INCLUDE (ISL). The initial plan was to develop and test various model architectures to determine the most effective for sign language knowledge transfer tasks. However, this objective was later abandoned to reduce the scope to a more feasible level.

Model Architecture

After identifying the most relevant landmarks and cleaning the dataset ac-

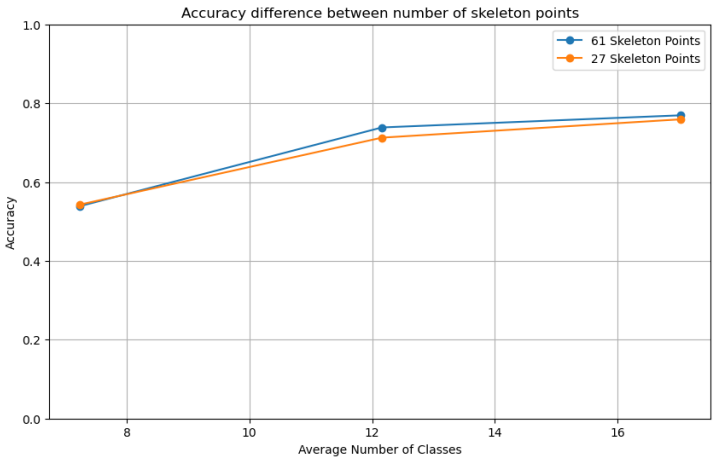


Figure 5.15: Accuracy of models trained on different skeleton points under different training sizes

cordingly, a more expressive model was trained to capture temporal dependencies and complex relationships in sign language sequences. While initial experiments utilized simple architectures like LSTM, a transition was made to a Transformer-based model Vaswani et al. (2017) due to its strong performance on sequential tasks and its ability to model long-range dependencies efficiently.

The Transformer architecture was inspired by the source code of Sridhar et al. (2020) but was rebuilt entirely using TensorFlow *tensorflow* (2025) to align with the study's preprocessing and integration requirements. The complete architecture is shown in Figure 5.16.

The model processed input sequences of shape `(batch_size, seq_length, num_features)`, where each frame contained concatenated landmark coordinates from the pose, left hand, and right hand. A dense layer projected these input features to a fixed hidden dimension of 128, followed by a learnable positional embedding layer that encoded the temporal order of frames.

The core of the model consists of 4 stacked Transformer encoder layers, each containing 8 multi-head self-attention heads. These layers help the model focus on the most relevant parts of the sequence and learn context-aware representations of gestures. Each Transformer encoder includes two sublayers: a multi-head attention mechanism followed by a feedforward dense block, and each of these is followed by layer normalization to ensure training stability and improve convergence.

Once the sequence is encoded, a Global Max Pooling layer is applied across the time dimension to summarize the sequence into a single vector representation. This is followed by a Dropout layer with a dropout rate of 0.3 to prevent overfitting, and finally, a dense output layer projects the pooled features to class logits corresponding to the number of sign language gesture classes in the dataset.

This architecture was compact yet expressive enough to model subtle differences in gesture trajectories, and its modular nature allowed us to easily test different configurations during ablation studies.

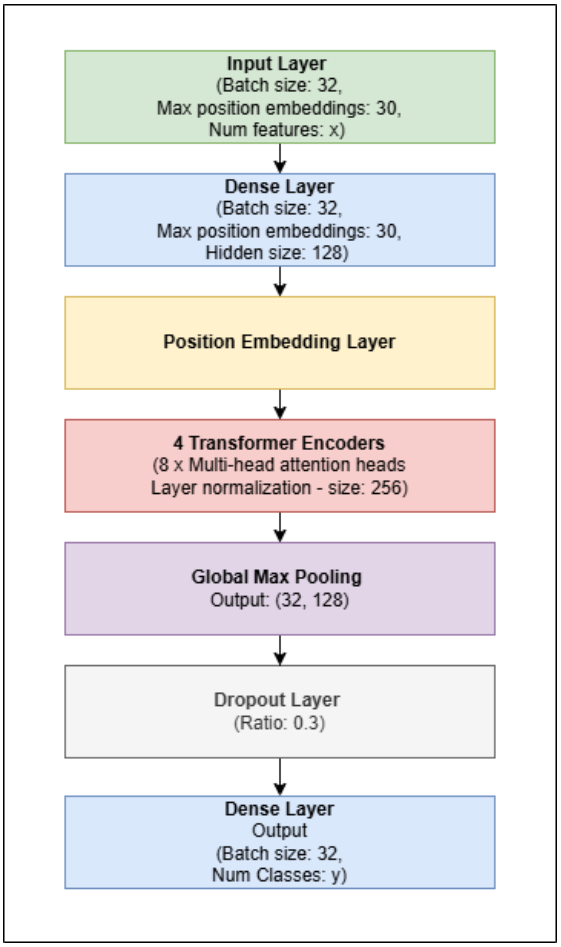


Figure 5.16: Transformer model architecture diagram

Data Loading and Preparation

The INCLUDE dataset's landmark data was loaded from preprocessed files. Each video was processed to a fixed length of 30 frames, using only the previously selected landmark points. For sequences exceeding 30 frames, frames were sampled to represent the entire gesture, avoiding overfitting as noted by Haputhanthri et al. (2022). However, the model's reliance on a fixed 30-frame input limits its ability to handle variable-length sequences.

The dataset was filtered to focus on top classes by instance count. We tested configurations with 80, 120, 160, 200, and 240 classes to evaluate model scalability and fine-tuning performance. Separately, we prepared a distinct 80-class dataset that prioritized a predefined set of overlapping words found previously in table 5.2. If any overlapping word was not among the top 80 classes, it was included

regardless (This was not needed for other model size like 120 and 160 because these classes were in those by default). Labels for all configurations were encoded in one-hot format. The data was split into training (60%), validation (16%), and test (24%) sets using a 60-40 split, followed by a 40-60 split of the remaining data. To enhance robustness, we applied small data augmentation to the training set by adding random noise (standard deviation 0.01). This simulated minor variations in landmark positions, such as slight hand or body shifts, while keeping validation and test sets unchanged.

Training Configuration

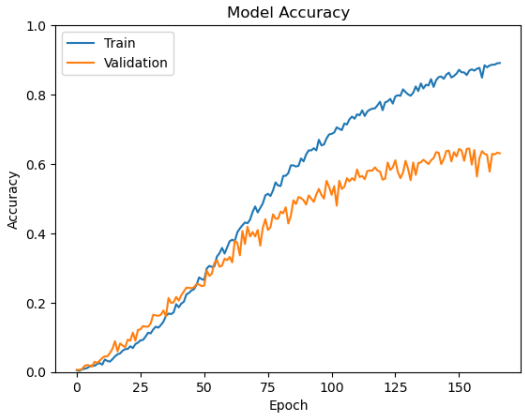
Transformer models were trained for each class configuration (80, 120, 160, 200, 240, and the separate 80-class overlapping set). Each model had a hidden size of 128, four encoder layers, and eight attention heads, as described previously. The models were compiled with the Adam optimizer (learning rate 10^{-4}), categorical cross-entropy loss with label smoothing (0.1) for improved generalization, and accuracy as the primary metric. To address class imbalance, balanced class weights were applied to prioritize underrepresented gestures. Early stopping was implemented, monitoring validation loss with a patience of 10 epochs, to prevent overfitting and restore the best weights. The data augmentation supported these settings, enabling the models to learn patterns robust to minor gesture variations. As for the training hardware, laptop computer with Intel Core i7 10870H and a GTX 1660Ti was used.

Training accuracies and training history were shown in Figures 5.17 and 5.18 , while testing accuracies were included in Table 5.4. Notably, F1 values could not be recorded for these models, as these experiments were conducted during initial phases.

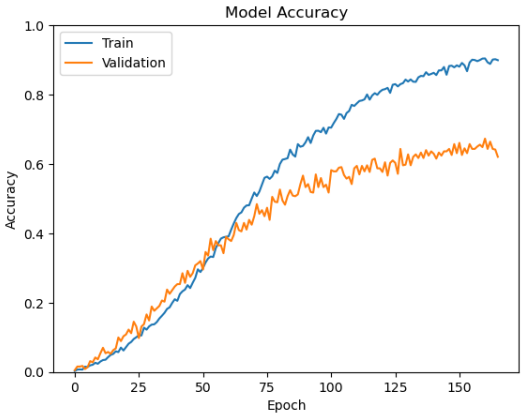
Model Configuration	**Test Accuracy**
80-Class ISL Base Model	0.6468
80-Class ISL Base Model (Meaning Overlapping with SSL)	0.6206
80-Class ISL Base Model (Movement Overlapping with SSL)	0.6905
120-Class ISL Base Model	0.6173
160-Class ISL Base Model	0.6478
200-Class ISL Base Model	0.6247
240-Class ISL Base Model	0.6079

Table 5.4: Test accuracies for Transformer models trained on different class configurations of the INCLUDE dataset.

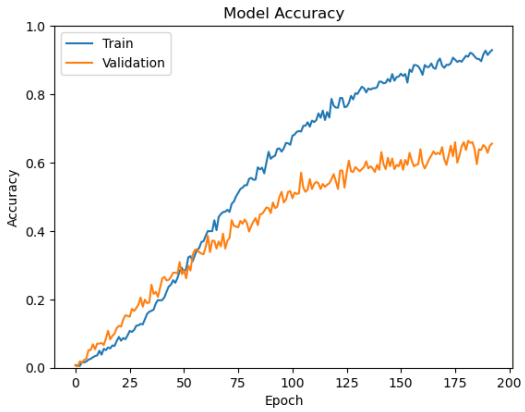
Overlapping Models



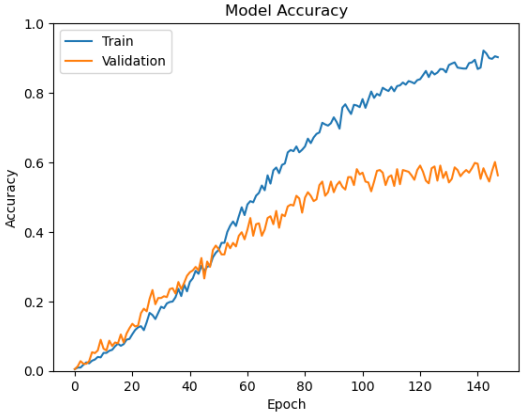
(a) 240-class



(b) 200-class



(c) 160-class

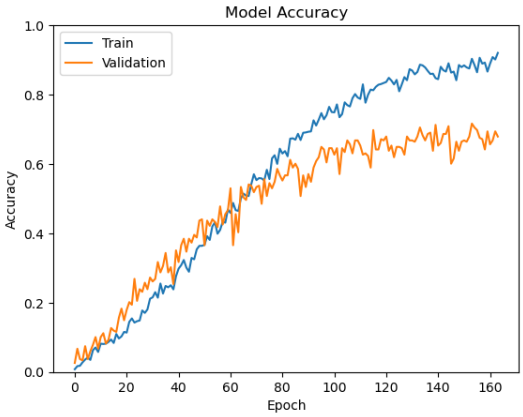


(d) 120-class

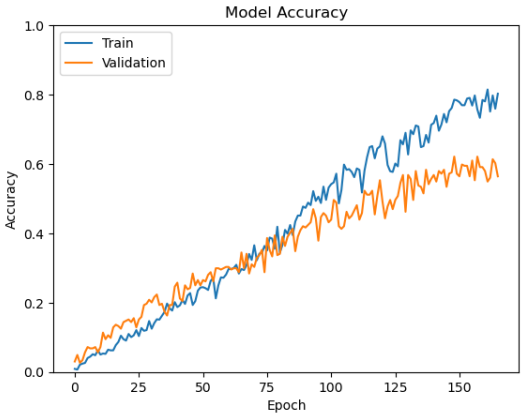
Figure 5.17: Training and validation performance of Base models for the 240, 200, 160, and 120-class configurations. HQ images can be accessed here

In addition to the six models described previously, an additional model, referred to as the "80-Class ISL Base Model (Movement Overlapping with SSL)" was included in Figure 5.18c and Table 5.4. This model was trained after the other six models. Initially, the "80-Class ISL Base Model (Meaning Overlapping with SSL)" was developed, incorporating glosses from the INCLUDE dataset that were also present in the SSL dataset, as identified in the semantic similarity analysis. Subsequently, it was determined that comparing sign movements between the two datasets could identify similar gestures. However, this proved challenging due to the need for 240 * 64 comparisons between ISL and SSL words.

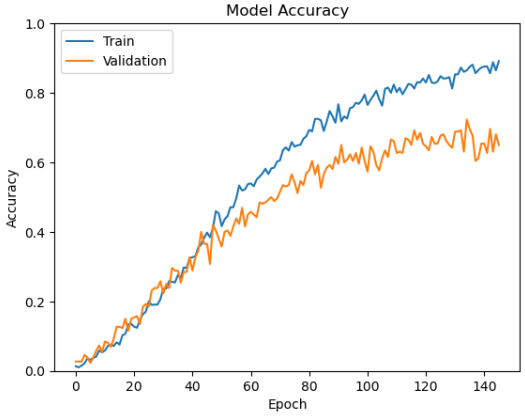
To address this, an attempt was made to compare signs based on their landmark trajectories using the Dynamic Time Warping (DTW) algorithm *DWT* (2025). This algorithm was capable of comparing time-series data (skeleton points per frame, treated as a time-series dataset) and outputting a similarity score. The plan was to select ISL-SSL word pairs with the highest similarity scores to train



(a) 80-class



(b) 80-class with overlapping meaning



(c) 80-class with movement overlapping

Figure 5.18: Training and validation performance of Base models for the 80-class variants. HQ images can be accessed here

a movement-overlapping model. However, this approach yielded insufficiently reliable results because even the different instances of the same sign word were having the different attributes and not the exactly the same (so it was difficult to compare).

Instead, the pre-trained models were leveraged. The six models, trained on the 240-gloss INCLUDE dataset as described previously, achieved approximately 60% accuracy on test data, classifying any sign language sequence into one of 240 ISL words. By inputting SSL data into these models, each SSL sequence was matched to the most similar ISL word based on movement patterns. This approach was precisely what was required for identifying movement-based overlaps. Consequently, each SSL datapoint was processed to determine its closest ISL match. However, the results were inconsistent. For example, 20 instances of the word "Good" (හොඳයි) in SSL were tested, and 10 instances were matched to the ISL word "Chair," with the remaining instances receiving other matches. To address

this, all 64 SSL words were sorted by the frequency of consistent ISL matches, and the top 20 words with the highest consistency were selected as movement-overlapping words, regardless of their semantic equivalence. Using these words, the "80-Class ISL Base Model (Movement Overlapping with SSL)" was trained, achieving a training accuracy of 0.6905. Table 5.5 presents the matched words in both datasets based on this criterion, while the process's flow was shown in Figure 5.19.

SSL Word	**INCLUDE Word**	**Instance Count**	**Number of Matches**
Thank_You	T-Shirt	20	19
Go	War	14	18
Talk?	Train	21	17
Can?	Afternoon	14	15
Little	Chair	14	14
Read	Bird	25	13
English	Animal	20	12
Good	wide	21	12
Still	new	21	12
Sinhala	Science	14	11
Start	Fish	20	8
Dialog box	healthy	21	9
Children	Newspaper	14	8
Video	Year	15	8
Hello	I	21	7
Children	God	14	7
House	old	21	7
See	Pleased	21	7
It	Bathroom	14	7
No	fast	21	5

Table 5.5: Words in SSL that have similar movement pattern to a some word in INCLUDE dataset. Instance count refers number of instances in INCLUDE dataset for. Number of Matches shows how many instances in SSL dataset matched with corresponding word in INCLUDE. Ex: Word "Thank You" in SSL dataset has somewhat similar sign movement to words "T-shirt" in INCLUDE dataset. And there are 20 instances from this word "T-shirt" in the INCLUDE dataset. When 20 instances from word "Thank you" are fed into the model, out of that 20, 19 of them were recognized as "T-shirt" by this model.

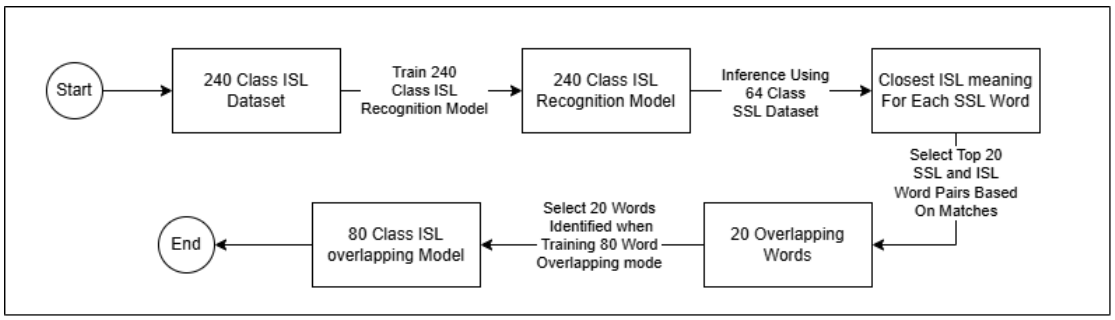


Figure 5.19: Flow chart of development process of movement overlapping model

5.2.6 Fine-tunning with SSL Dataset

Once all seven base models were trained for ISL, the next step was to investigate fine-tuning for SSL. Prior to fine-tuning, a baseline accuracy for SSL was established to evaluate the potential benefits of transfer learning for low-resourced sign languages. The selected low-resourced sign language was SSL, with a dataset comprising 64 words, each with approximately 50 instances, as described previously. However, 50 instances per word indicated that this dataset was not truly low-resourced, as training a word classification model with acceptable accuracy using such a large number of examples was straightforward. Figure 5.20 presents the training history for a model trained on this dataset, utilizing the same Transformer architecture described earlier. Validation accuracy exceeded 80% with relative ease. For the testing set (unseen data), the model achieved an accuracy of 81.15% and an average F1 score of 0.81 across three separate training sessions.

In the true low-resourced language, researchers do not have access to to this kind of large number of instances per class in datasets. Even this dataset, Alwis (2023) was not able to collect all these instance using real signers. Instead they have collected about half of the instance in each word using real signers and filled the rest by performing signs by researchers who did that study. Now let us say there was a requirement to collect another 100 SSL words. Now this would again a really complicated task to do. However, it is still possible to collect small number of instance from each class using 1 or 2 signers. Therefore to continue this study with more realistic scenario it was decided to only select a subset of instance from each word to train models. Therefore 2, 3, 4 or 6 were used as the number of instances from each class to train models to simulate low resource scenarios while testing and validation sets had 7 instance per class.

Baseline: Low Resource without Fine-tuning

To establish a reference for fine-tuning experiments, performance metrics were collected for Transformer models trained directly on the SSL dataset without pre-

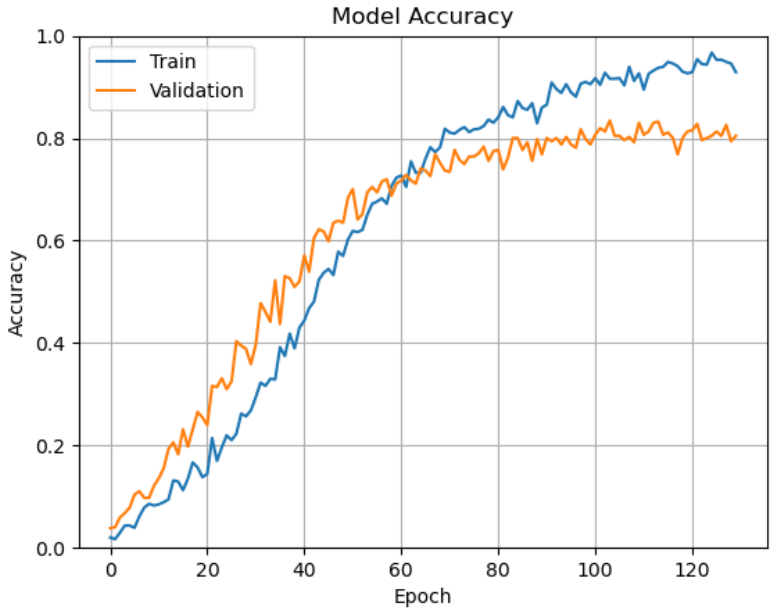


Figure 5.20: Transformer network trained with full SSL dataset

trained weights, termed Direct Models. These models utilized the same architecture and training configurations described in the pre-training section. Experiments varied the number of training instances per class (2, 3, 4, or 6) and the number of classes (48 or 64) to assess their impact on performance. Due to high variability with low instance counts, each configuration was executed five times, with averages and standard deviations reported for accuracy, F1 score, precision, recall, and training epochs.

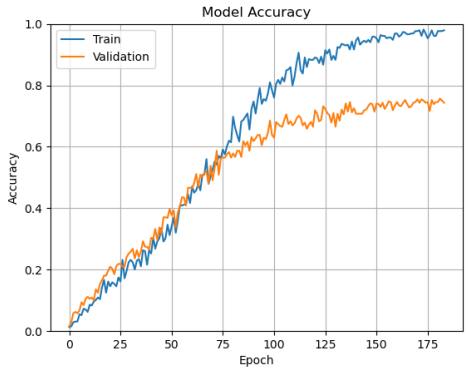
The SSL dataset comprised video sequences with landmark data capturing key points on the body and hands. Sequences were standardized to 30 frames by evenly sampling available frames or using all frames if fewer than 30 existed, with missing landmarks assigned (0,0) coordinates. The 27 pose and hand indices selected previously (0, 15–20 for pose; 0, 4, 7, 8, 11, 12, 15, 16, 19, 20 for each hand) were used to focus on relevant features, ensuring a consistent input format across experiments. TensorFlow's default splitting apportioned data proportionally across the entire dataset, not per class. For example, a 0.5 training and 0.5 validation split on a dataset with 64 classes and 20 instances each did not guarantee 10 instances per class in the training set, only that the training set contained $(64 \times 20) \times 0.5$ total data points. total data points. To address this, a custom function ensured each class in the SSL dataset had exactly 2, 3, 4, or 6 training instances, based on the experiment configuration. For fairness, the testing dataset included 7 instances per class, regardless of other parameters. This data preparation and splitting approach was replicated in fine-tuning experiments to ensure

comparability.

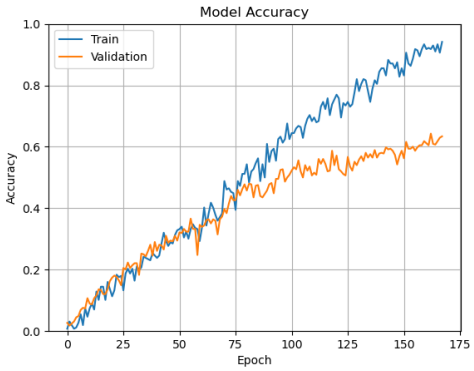
For the run with median performance in each configuration, training and validation accuracy plots and confusion matrices were generated to analyze model behavior. Performance metrics across all configurations are presented in Table 5.6, which highlights the challenges of training with limited data and provides a benchmark for comparing Direct and fine-tuned models.

Classes	**IPC**	**Accuracy**	**F1 Score**	**Epochs (± std)**
64	6	0.7679 ± 0.0156	0.7634 ± 0.0148	152.60 ± 17.59
64	4	0.6687 ± 0.0239	0.6636 ± 0.0241	174.20 ± 15.04
64	3	0.5277 ± 0.0238	0.5237 ± 0.0213	185.80 ± 11.11
64	2	0.3241 ± 0.0405	0.3166 ± 0.0405	168.20 ± 21.82
48	6	0.7970 ± 0.0167	0.7897 ± 0.0166	155.60 ± 16.74
48	4	0.6369 ± 0.0178	0.6322 ± 0.0199	165.20 ± 16.23
48	3	0.5375 ± 0.0229	0.5397 ± 0.0207	177.60 ± 18.86
48	2	0.2577 ± 0.1129	0.2353 ± 0.1218	110.80 ± 60.99

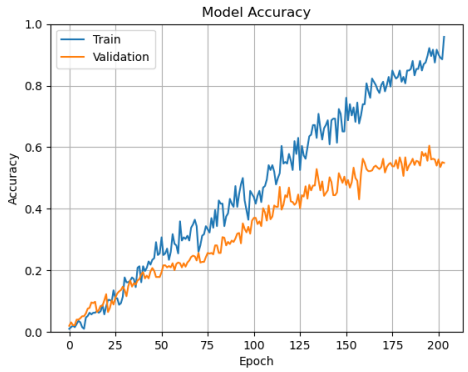
Table 5.6: Performance metrics (accuracy, F1 score, and training epochs before early stopping kicked in) of SSL models with varying class and instance configurations. Values are averaged over 5 runs with standard deviation shown. IPC represents number of training instances per class.



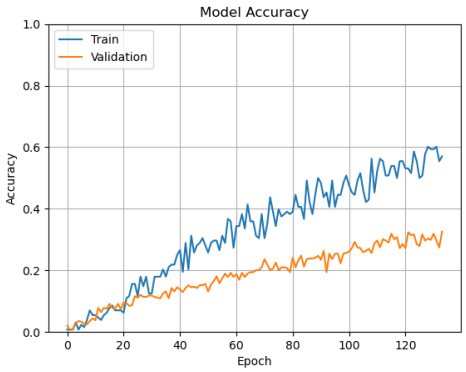
(a) 64-class, 6-instance



(b) 64-class, 4-instance

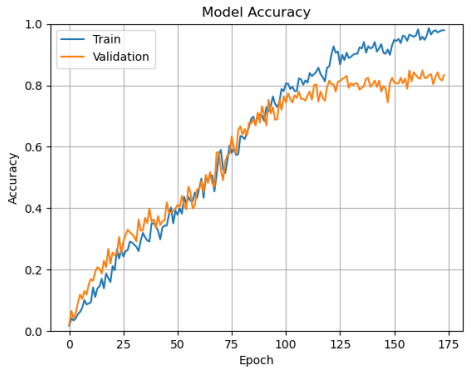


(c) 64-class, 3-instance

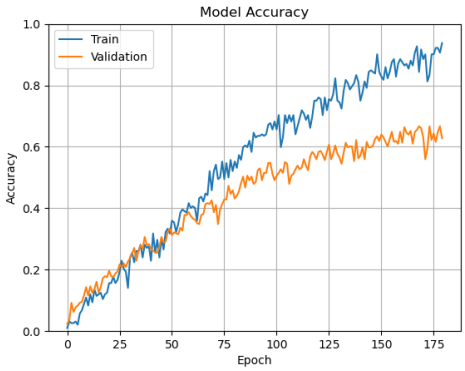


(d) 64-class, 2-instance

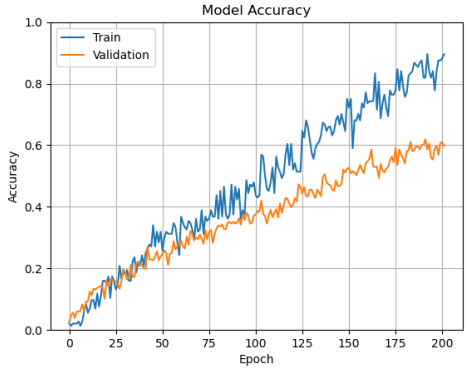
Figure 5.21: Training and validation performance of SSL models for 64-class with varying instance configurations. HQ images can be accessed here



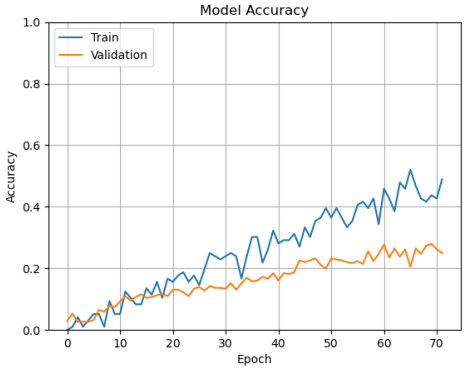
(a) 48-class, 6-instance



(b) 48-class, 4-instance



(c) 48-class, 3-instance



(d) 48-class, 2-instance

Figure 5.22: Training and validation performance of SSL models for 48-class with varying instance configurations. HQ images can be accessed here

Training plots for all eight Direct Models are presented in Figure 5.21 and 5.22 . It was observed that accuracy decreased as the number of instances per class reduced. Despite techniques to mitigate overfitting (e.g., dropout, data augmentation, early stopping), considerable overfitting was evident in all models. A clearer understanding of model performance on the training set was obtained through confusion matrices. However, including readable 48x48 and 64x64 confusion matrices in this document was impractical. These matrices were therefore provided in a separate Google Drive folder, accessible via links in Appendix A.

Fine-tuning the Base Models

To adapt pre-trained Transformer models for the SSL dataset, a fine-tuning approach was employed, leveraging features learned from the larger ISL dataset. These base models, pre-trained on ISL with varying class counts (e.g., 80, 120), had captured robust temporal and spatial features from sign language data. The objective was to investigate how transfer learning could improve performance in

low-resource settings, such as SSL, where training data per class was limited.

The fine-tuning process mirrored that used for training the Direct Models to ensure fairness. New Transformer model instances were initialized, and pre-trained ISL weights were loaded, maintaining compatibility with the original architecture. To adapt to SSL, the final dense layer was replaced with a new layer matching the SSL class count (48 or 64), initialized with He normalization and L2 regularization to reduce overfitting. All layers, including those from the pre-trained base, were set to trainable, enabling the model to adjust ISL-learned features to SSL-specific patterns.

Training settings from the pre-training and Direct Model phases were reused for consistency. The Adam optimizer was employed with a learning rate of 1×10^{-4} for stable convergence, and categorical cross-entropy with label smoothing (0.1) was used as the loss function to reduce overconfidence. Random noise with a standard deviation of 0.01 was applied to training sequences to curb overfitting. Early stopping was configured to halt training if validation loss did not improve for 10 epochs, restoring the best weights. Models were trained for up to 250 epochs with a batch size of 32, though early stopping often intervened earlier.

The experimental design involved fine-tuning seven base models under controlled conditions, testing with 2, 3, 4, or 6 training instances per class to evaluate performance in low-data scenarios. The number of classes was varied, using SSL datasets with 48 and 64 classes to assess the impact of class size. Each configuration was executed five times to account for variability, with performance metrics—accuracy, F1 score, precision, and recall—averaged across runs. For the run with median performance in each configuration, training and validation accuracy plots and confusion matrices were generated to analyze model behavior.

This setup enabled direct comparison with models trained from scratch on SSL, highlighting the benefits of transfer learning. It was hypothesized that fine-tuned models, leveraging ISL's broad feature exposure, would outperform Direct Models, particularly with fewer training instances (e.g., 2, 3, 4, or 6 per class). Results in Tables 5.7 and 5.8 supported this hypothesis in some cases. Initial analysis indicated an average 8% accuracy increase across all configurations, demonstrating that the Transformer's ability to model temporal dependencies, refined through ISL pre-training, enhanced generalization on SSL's limited data. Insights into the effects of class size (48 vs. 64 classes) and performance stability across runs will be detailed in the Results Analysis and Critical Evaluation section.

Model Description	IPC	Epochs	F1 Score	Accuracy	Direct Acc.	Diff
240 → 64	6	63.6 (±15.5)	0.7595 (±0.0216)	0.7616 (±0.0188)	0.7679	-0.63%
200 → 64	6	84.2 (±18.8)	0.7595 (±0.0207)	0.7670 (±0.0189)	0.7679	-0.09%
160 → 64	6	73.0 (±10.2)	0.7498 (±0.0215)	0.7522 (±0.0224)	0.7679	-1.57%
120 → 64	6	82.8 (±17.5)	0.7408 (±0.0292)	0.7429 (±0.0274)	0.7679	-2.5%
80 → 64	6	101.4 (±11.8)	0.7739 (±0.0172)	**0.7830 (±0.0134)**	0.7679	**1.51%**
80 (Overlap) → 64	6	103.2 (±4.8)	0.7465 (±0.0125)	0.7536 (±0.0128)	0.7679	-1.43%
80 (Movement Overlap) → 64	6	90.4 (±9.1)	0.7421 (±0.0195)	0.7455 (±0.0180)	0.7679	-2.24%
240 → 64	4	66.8 (±7.4)	0.6484 (±0.0284)	0.6612 (±0.0296)	0.6687	-0.75%
200 → 64	4	82.0 (±6.6)	0.6447 (±0.0272)	0.6536 (±0.0273)	0.6687	-1.51%
160 → 64	4	101.0 (±19.0)	0.6572 (±0.0113)	0.6674 (±0.0123)	0.6687	-0.13%
120 → 64	4	76.6 (±14.5)	0.6179 (±0.0250)	0.6196 (±0.0278)	0.6687	-4.91%
80 → 64	4	95.6 (±13.6)	0.6705 (±0.0130)	**0.6786 (±0.0136)**	0.6687	**0.99%**
80 (Overlap) → 64	4	92.4 (±6.9)	0.6246 (±0.0151)	0.6330 (±0.0133)	0.6687	-3.57%
80 (Movement Overlap) → 64	4	106.4 (±11.6)	0.6426 (±0.0269)	0.6491 (±0.0268)	0.6687	-1.96%
240 → 64	3	72.8 (±18.4)	0.5700 (±0.0216)	**0.5741 (±0.0242)**	0.5277	**4.64%**
200 → 64	3	78.8 (±13.3)	0.5683 (±0.0176)	**0.5710 (±0.0182)**	0.5277	**4.33%**
160 → 64	3	90.6 (±14.6)	0.5722 (±0.0148)	**0.5790 (±0.0115)**	0.5277	**5.13%**
120 → 64	3	72.8 (±8.4)	0.5316 (±0.0228)	**0.5393 (±0.0221)**	0.5277	**1.16%**
80 → 64	3	81.0 (±11.6)	0.5562 (±0.0151)	**0.5616 (±0.0162)**	0.5277	**3.39%**
80 (Overlap) → 64	3	96.0 (±13.3)	0.5563 (±0.0179)	**0.5732 (±0.0187)**	0.5277	**4.55%**
80 (Movement Overlap) → 64	3	99.8 (±8.1)	0.5530 (±0.0159)	**0.5634 (±0.0143)**	0.5277	**3.57%**
240 → 64	2	68.0 (±12.7)	0.4407 (±0.0284)	**0.4576 (±0.0278)**	0.3241	**13.35%**
200 → 64	2	76.8 (±9.8)	0.4291 (±0.0163)	**0.4348 (±0.0186)**	0.3241	**11.07%**
160 → 64	2	76.0 (±3.6)	0.4225 (±0.0095)	**0.4290 (±0.0121)**	0.3241	**10.49%**
120 → 64	2	71.6 (±7.2)	0.4492 (±0.0189)	**0.4576 (±0.0195)**	0.3241	**13.35%**
80 → 64	2	86.6(±14.9)	0.4793 (±0.0126)	**0.4879 (±0.0178)**	0.3241	**16.38%**
80 (Overlap) → 64	2	93.4 (±15.2)	0.4185 (±0.0410)	**0.4290 (±0.0397)**	0.3241	**10.49%**
80 (Movement Overlap) → 64	2	87.2 (±4.9)	0.3960 (±0.0116)	**0.4138 (±0.0126)**	0.3241	**8.97%**

Table 5.7: Finetuned Model Evaluation Results for 64 SSL Words with Highlighted Accuracy Improvements

Model Description	IPC	Epochs	F1 Score	Accuracy	Direct Acc.	Diff
240 → 48	6	74.4 (±4.1)	0.7633 (±0.0120)	0.7750 (±0.0127)	0.7970	-2.2%
200 → 48	6	76.8 (±9.2)	0.7692 (±0.0116)	0.7720 (±0.0099)	0.7970	-2.5%
160 → 48	6	74.2 (±14.7)	0.7557 (±0.0117)	0.7601 (±0.0133)	0.7970	-3.69%
120 → 48	6	78.2 (±9.1)	0.7617 (±0.0164)	0.7673 (±0.0104)	0.7970	-2.97%
80 → 48	6	74.6 (±10.6)	0.7860 (±0.0167)	0.7923 (±0.0150)	0.7970	-0.47%
80 (Overlap) → 48	6	89.8 (±14.3)	0.7497 (±0.0137)	0.7565 (±0.0107)	0.7970	-4.05%
80 (Movement Overlap) → 48	6	90.6 (±7.0)	0.7738 (±0.0261)	0.7780 (±0.0233)	0.7970	-1.9%
240 → 48	4	61.0 (±7.0)	0.6690 (±0.0330)	**0.6732 (±0.0303)**	0.6369	**3.63%**
200 → 48	4	77.8 (±15.0)	0.6915 (±0.0221)	**0.7042 (±0.0202)**	0.6369	**6.73%**
160 → 48	4	76.0 (±8.7)	0.6575 (±0.0190)	**0.6589 (±0.0181)**	0.6369	**2.2%**
120 → 48	4	89.0 (±12.3)	0.6376 (±0.0246)	**0.6524 (±0.0222)**	0.6369	**1.55%**
80 → 48	4	81.6(±11.5)	0.6749 (±0.0138)	**0.6810 (±0.0143)**	0.6369	**4.41%**
80 (Overlap) → 48	4	103.4 (±23.5)	0.6834 (±0.0180)	**0.6887 (±0.0172)**	0.6369	**5.18%**
80 (Movement Overlap) → 48	4	105.4 (±21.1)	0.6531 (±0.0303)	**0.6631 (±0.0238)**	0.6369	**2.62%**
240 → 48	3	65.0 (±12.0)	0.5983 (±0.0255)	**0.6077 (±0.0250)**	0.5375	**7.02%**
200 → 48	3	64.6 (±12.1)	0.5979 (±0.0161)	**0.5988 (±0.0191)**	0.5375	**6.13%**
160 → 48	3	82.8 (±13.0)	0.6276 (±0.0207)	**0.6387 (±0.0205)**	0.5375	**10.12%**
120 → 48	3	74.6 (±15.6)	0.5709 (±0.0197)	**0.5768 (±0.0188)**	0.5375	**3.39%**
80 → 48	3	87.4 (±7.6)	0.6307 (±0.0107)	**0.6387 (±0.0128)**	0.5375	**10.12%**
80 (Overlap) → 48	3	81.4 (±5.2)	0.5476 (±0.0136)	**0.5631 (±0.0161)**	0.5375	**2.56%**
80 (Movement Overlap) → 48	3	96.0 (±14.1)	0.5611 (±0.0188)	**0.5679 (±0.0246)**	0.5375	**3.04%**
240 → 48	2	62.2 (±6.0)	0.5059 (±0.0207)	**0.5071 (±0.0211)**	0.2577	**24.94%**
200 → 48	2	63.2 (±17.1)	0.4800 (±0.0376)	**0.4821 (±0.0381)**	0.2577	**22.44%**
160 → 48	2	71.4 (±16.6)	0.4247 (±0.0255)	**0.4310 (±0.0260)**	0.2577	**17.33%**
120 → 48	2	63.8 (±10.3)	0.4594 (±0.0269)	**0.4732 (±0.0260)**	0.2577	**21.55%**
80 → 48	2	76.0 (±14.2)	0.5225 (±0.0250)	**0.5327 (±0.0204)**	0.2577	**27.5%**
80 (Overlap) → 48	2	90.4 (±13.6)	0.4558 (±0.0254)	**0.4649 (±0.0256)**	0.2577	**20.72%**
80 (Movement Overlap) → 48	2	87.6 (±23.0)	0.4361 (±0.0498)	**0.4476 (±0.0429)**	0.2577	**18.99%**

Table 5.8: Finetuned Model Evaluation Results for 48 SSL Words with Highlighted Accuracy Improvements

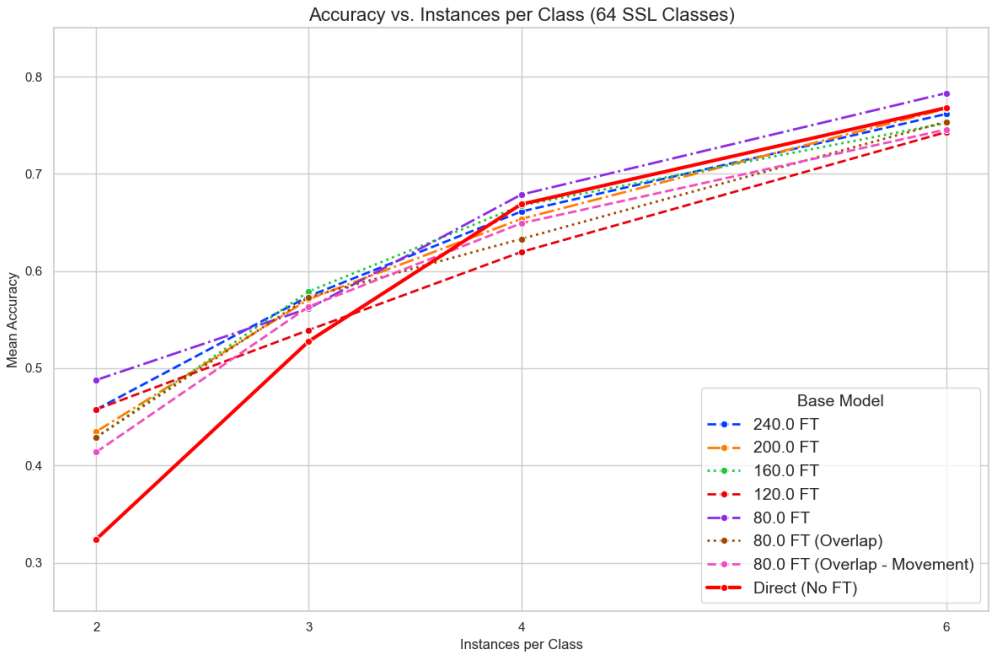
6 Result Analysis and Critical Evaulation

A total of 64 models were trained to recognize words in SSL. Eight of these were Direct Models, trained directly on SSL data, while the remaining models were pre-trained on the larger ISL dataset and fine-tuned to recognize SSL words. This section analyzed the results through visualizations to identify underlying patterns derived from the extensive experiments conducted during the study.

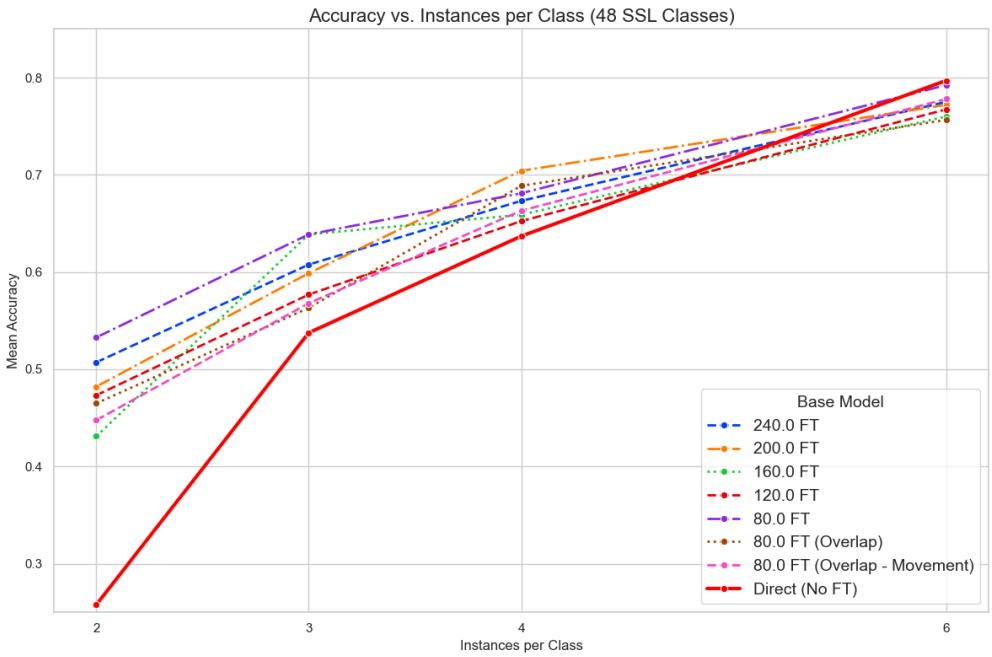
6.0.1 Accuracy Comparison Between Direct Models and Fine-Tuned Models

Two line plots were generated for the 64- and 48-class models to evaluate accuracy gains from transfer learning techniques. These are presented in Figure 6.23. It was observed that, with very low training instance counts (2 or 3), fine-tuned models performed significantly better than Direct Models, regardless of the base model size or the number of classes in the SSL dataset. However, this performance gap diminished with higher instance counts, particularly for the 64-class models with 4 or 6 instances per class. These plots indicated that the hypothesis—cross-lingual transfer learning improves performance for low-resourced languages like Sinhala—was supported only for very low instance counts (2–3). When the number of instances per class reached 4 or 6, transfer learning yielded minimal improvement.

This reduction in performance gap could be attributed to several factors: the Direct Models may have sufficient data at these instance counts to learn SSL-specific patterns effectively, reducing the reliance on pre-trained ISL features; the Transformer architecture's capacity to model temporal dependencies might be fully utilized with 4 or 6 instances, minimizing the advantage of transfer learning; or the similarity between ISL and SSL movement patterns may not provide additional discriminative power for SSL-specific signs when more data is available.



(a) 64 class SSL models HQ Image



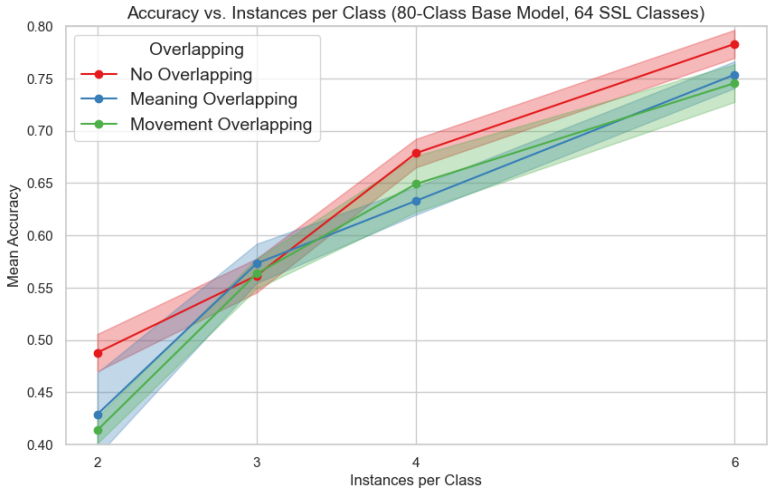
(b) 48 class SSL models HQ Image

Figure 6.23: Testing accuracy comparison between direct models and fine-tuned models for all the instance numbers and base models. Red color lines represents the direct models

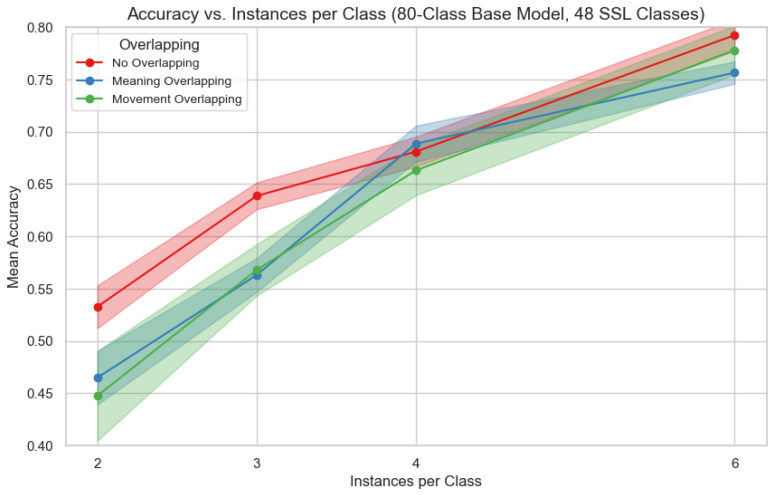
6.0.2 Effectiveness of Overlapping Models

Among the 80-class base models, one model was trained with the top ISL words by instance count, and two models were trained with words similar to SSL in meaning

or sign movement, overriding the instance-count rule. Figure 6.24 illustrates the impact of overlapping models on different SSL class sizes. It was observed that, in both graphs, the non-overlapping base model generally achieved higher accuracy than the overlapping models in most cases. Although it appeared that overlapping words provided limited benefits for fine-tuning, this outcome may have resulted from the non-overlapping model being pre-trained with a slightly larger number of training examples (because some overlapping words were not in top 80 most frequent words in INCLUDE). Consequently, it was concluded that the impact of training with words similar in meaning or sign movement on improving transfer learning capabilities in SLR could not be definitively determined.



(a) 64 class SSL fine-tuning HQ Image



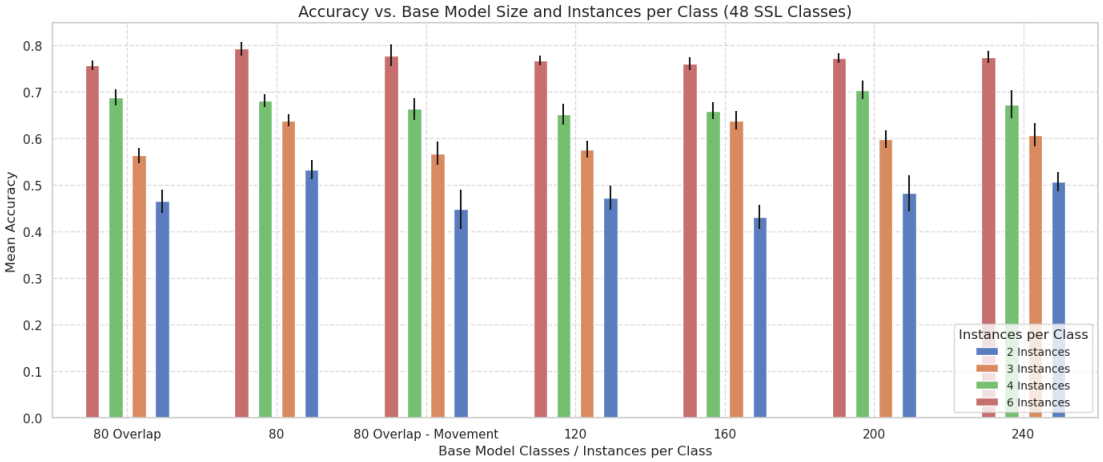
(b) 48 class SSL fine-tuning HQ Image

Figure 6.24: Testing accuracy comparison between overlapping models and non-overlapping models. Shaded area shows the standard deviation of accuracy values.

6.0.3 Impact of Base Model Size

A total of seven base models were tested during the study. Three of these had the same size (80 classes), while the remaining four utilized different numbers of ISL classes (120, 160, 200, 240). Figure 6.25 illustrates how base model size affected fine-tuning performance across various instance counts (2, 3, 4, 6) and class sizes (48, 64) in SSL. It was evident that base model size, within the tested range, had no consistent impact on fine-tuning performance. This could mean that even with a small number of words in primary language (ISL) can help to model to learn good amount of features that can help to learn SSL features better later. Positive side of this discovery is that when further studying this area, one could easily train base models without spending much time on that.

(a) 64 class HQ Image



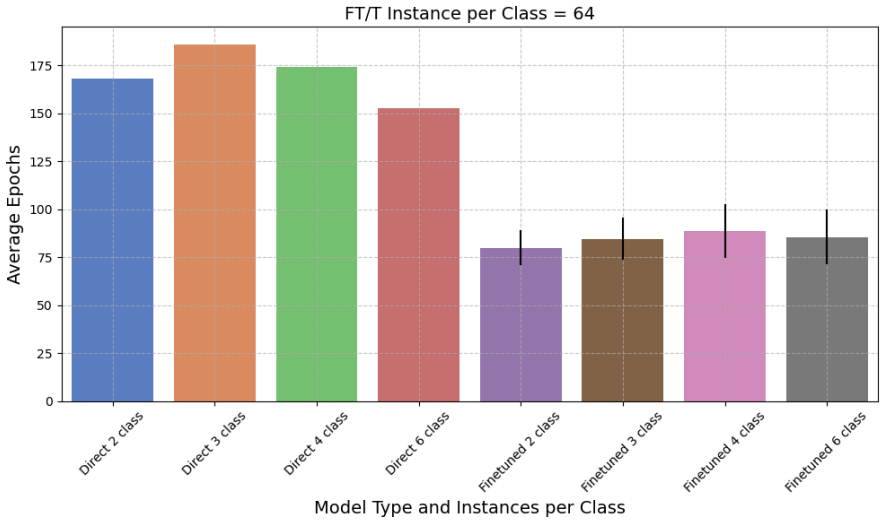
(b) 48 class HQ Image

Figure 6.25: Test accuracy comparison between different sized base models. Dark line on top of each bar represents the standard deviation of multiple runs of the same model.

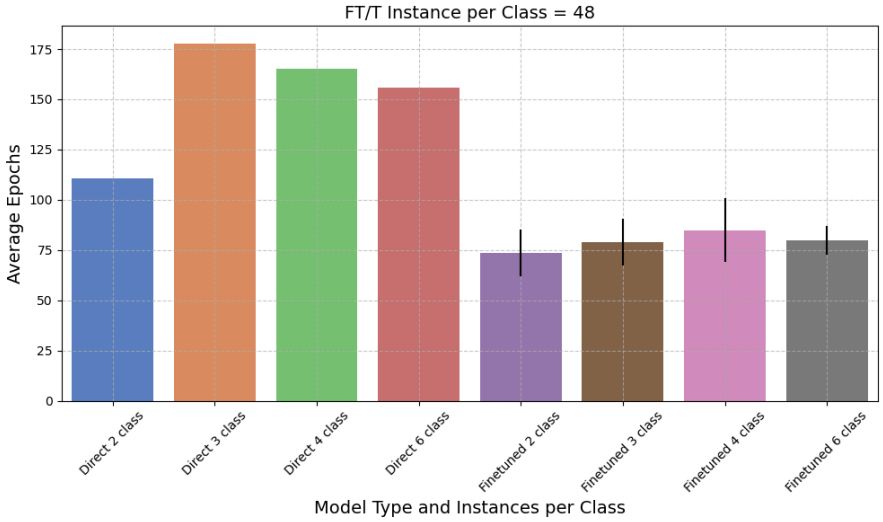
6.0.4 Comparison of Training Epochs Before Early Stopping

This study utilized varying amounts of data to simulate low-resource scenarios. To mitigate overfitting, early stopping techniques were employed. Consequently, the number of epochs required for model convergence was tracked. Figure 6.26 presents the average number of training epochs for each model trained. For each instance count (2, 3, 4, 6), plots reflect the average epochs across all base models (e.g., 80, 80 overlapping, 120, 160, 200, 240) fine-tuned with the specified instance count. Standard deviations were included in the bar charts (indicated by black lines) to account for variability among fine-tuned models. As only one variant of each Direct Model was trained per instance count, standard deviations were not included for these.

It was evident that, despite using the same number of parameters, fine-tuned models consistently required fewer epochs to converge compared to Direct Models. This indicated that fine-tuned models had partially learned temporal and spatial patterns during the pre-training phase, enabling faster convergence. While early convergence was not a direct metric to confirm the efficacy of cross-lingual fine-tuning in the SLR domain, it suggested that pre-training on a larger dataset facilitated pattern learning. Furthermore, the 48-class, 2-instance Direct Model exhibited an anomalously low number of epochs, as shown in Figure 6.26b. Multiple tests were conducted to verify this result, confirming its consistency. This anomaly was likely due to the model being trained on significantly fewer instances compared to other Direct Model (48 words, 2 instances per word), leading to faster weight convergence but lower overall testing accuracy.



(a) 64 class HQ Image



(b) 48 class HQ Image

Figure 6.26: Number of epochs models trained before early stopping kicked in and stopped the training process.

7 Discussion & Conclusion

This study aimed to investigate the efficacy of cross-lingual transfer learning in enhancing Sign Language Recognition (SLR) for Sinhala Sign Language (SSL), a low-resourced sign language with limited training data. The primary objectives were to evaluate whether pre-training on a high-resource language like Indian Sign Language (ISL) could improve SSL recognition accuracy, identify patterns and trends in transfer learning techniques, and determine the most suitable primary languages for pre-training, though the latter was constrained to ISL due to time limitations. Through comprehensive experiments, the research demonstrated that Transformer-based models, pre-trained on ISL and fine-tuned on small SSL datasets, significantly outperform models trained directly on SSL, particularly in extreme low-data scenarios (2–3 instances per class), achieving up to an 8% accuracy improvement.

These findings validate the potential of transfer learning to address data scarcity in SLR, offering a pathway for developing accessible communication systems for underrepresented sign languages like SSL. The following discussion analyzes the results in the context of the research questions, highlights key patterns, and evaluates the implications and limitations of the study, culminating in conclusions that underscore its contributions to the field.

Comprehensive experiments on cross-lingual transfer learning for SSL, a low-resourced language, have yielded valuable insights into accuracy improvements and the training dynamics of Transformer-based models. The study demonstrated significant performance gains when training instances per class were very low (2 or 3), with fine-tuned models achieving up to an 8% accuracy increase over Direct Models. This success highlights the potential of leveraging a high-resourced language like ISL to bootstrap SLR systems in data-scarce scenarios, particularly when only a few examples per class are available, as is common in real-world low-resource settings. Futhermore, this answers the first research question: **Is it possible to improve the accuracy of Sinhala Sign Language (SSL) models using cross-lingual transfer learning** with a "Yes". The simulation of data scarcity by limiting instances to 2–6 per class effectively mirrored the challenges of collecting large SSL datasets, underscoring the practical utility of transfer learning for rapid deployment of SLR systems with minimal data.

While answering the second research question: **What underlying patterns and trends can be uncovered through transfer learning techniques?**, it was possible to find that the transformer architecture's ability to model temporal dependencies and long-range relationships in sign language sequences was likely a key factor in these gains, with pre-trained ISL features enabling better generalization at low instance counts. However, the diminishing performance gap at

higher instance counts (4 or 6) suggests that Direct Models can effectively learn SSL-specific patterns with sufficient data, potentially saturating the model's capacity and reducing the advantage of transfer learning. The use of skeleton points extracted via MediaPipe proved computationally efficient and robust, mitigating biases from background or signer appearance, though it incurred a reported 4–6% accuracy trade-off compared to video-based inputs. The decision to use 27 selected points, which maintained performance while reducing complexity, indicates that careful feature selection is critical in low-resource settings, though integrating visual features could enhance future models.

Extending the answer for second research question, the number of epochs before early stopping provided further evidence of transfer learning's benefits, with fine-tuned models converging faster than Direct Models, reflecting pre-learned temporal and spatial patterns from ISL. This efficiency is particularly promising for resource-constrained environments, reducing computational demands on modest hardware like the laptop used in this study (Intel Core i7 10870H, GTX 1660Ti). However, overfitting remained a challenge, especially for Direct Models with low instance counts, despite techniques like dropout, data augmentation, and early stopping. This suggests that advanced regularization or tailored augmentation strategies for skeleton data could further improve robustness.

Some findings were surprising or inconclusive. The lack of consistent impact from varying base model sizes (80 to 240 classes) likely stems from data overlap in the ISL dataset, obscuring pre-training scale effects. Future work with non-overlapping datasets could clarify this. The unexpected underperformance of overlapping models, designed to leverage ISL-SSL similarities, suggests limitations in the similarity methods (DTW and pre-trained classifications).

As for the third research question: **What are the most suitable primary languages for pre-training a model to recognize Sinhala Sign Language (SSL)?**, due to time limitations, it was not possible to conduct all experiments in more than one language. Therefore, this question still remains unanswered and is yet to be discussed in future work.

7.1 Research Contributions and Novelty

This research contributes to the growing body of knowledge in SLR by being one of the first studies to systematically explore cross-lingual transfer learning for Sinhala Sign Language (SSL), a low-resourced sign language with very limited available datasets. The novelty of the work lies in demonstrating that Transformer-based models, pre-trained on a high-resource language such as Indian Sign Language (ISL), can significantly improve SSL recognition performance when training data is extremely scarce. This contribution is particularly impactful in settings where

collecting large-scale sign language datasets is impractical due to economic or logistical constraints.

This thesis sets a foundation for developing accessible, scalable, and efficient SLR systems for underrepresented sign languages like SSL. It not only bridges a critical research gap but also opens new directions for multilingual sign language recognition and inclusive communication technologies.

8 Limitations and Future Work

This study focused solely on SSL even though the topic was low-resourced languages. Given that different sign languages exhibit distinct movements, results may vary, and developing consistent multilingual models in this domain was challenging. The original plan included testing multiple primary languages for pre-training, but this was abandoned due to time constraints and limited generalizability, as evaluating only a few languages would not be comprehensive. Therefore, during this study it was not possible to answer the 3rd research question. Similarly, the initial proposal to assess fine-tuning performance across various deep learning architectures and techniques was deemed overly broad and thus excluded. Another key limitation was the focus on low-resourced methods for ISLR mode only. In more realistic settings, evaluating these methods in CSLR mode would be valuable.

There were unexpected results when it comes to comparison between overlapping models. As explained before, this could be due the poor techniques used to meassure similarties between two datasets. These could be replaced with advanced techniques, like deep learning-based embeddings, to better capture gesture similarities and improve transfer learning in future. Additionally, while the use of skeleton points was considered an advantage, it may also represent a limitation, as fine-tuning models that support visual inputs with larger datasets could leverage learned features for smaller datasets. For instance, pre-trained models like He et al. (2015) on large general datasets have been fine-tuned for specific tasks. These approaches could be explored in SLR tasks as future work.

References

Alwis, A. A. G. D. (2023), 'Making video conferencing accessible for the hearing-impaired community'.

Aly, S. & Aly, W. (2020), 'Deeparslr: A novel signer-independent deep learning framework for isolated arabic sign language gestures recognition', *IEEE Access* **8**, 83199–83212.

Bazarevsky, V., Grishchenko, I., Raveendran, K., Zhu, T., Zhang, F. & Grundmann, M. (2020), 'Blazepose: On-device real-time body pose tracking'.
URL: *http://arxiv.org/abs/2006.10204*

Bilge, Y. C., Ikizler-Cinbis, N. & Cinbis, R. G. (2019), 'Zero-shot sign language recognition: Can textual data uncover sign languages?'.
URL: *http://arxiv.org/abs/1907.10292*

Cicirelli, G. & D'Orazio, T. (2017), 'Gesture recognition by using depth data: Comparison of different methodologies', *Motion Tracking and Gesture Recognition* .

Coogan, T. & Sutherland, A. (2006), 'Transformation invariance in hand shape recognition'.

Coster, M. D., Rushe, E., Holmes, R., Ventresque, A. & Dambre, J. (2023), 'Towards the extraction of robust sign embeddings for low resource sign language recognition'.
URL: *http://arxiv.org/abs/2306.17558*

Dardas, N. H. & Georganas, N. D. (2011), 'Real-time hand gesture detection and recognition using bag-of-features and support vector machine techniques', *IEEE Transactions on Instrumentation and Measurement* **60**, 3592–3607.

DWT (2025). Accessed: 2, 2025.
URL: *https://en.wikipedia.org/wiki/Dynamic$_t$ime$_w$arping*

Elakkiya, R. (2021), 'Machine learning based sign language recognition: a review and its research frontier'.

Elmezain, M., Al-Hamadi, A. & Michaelis, B. (2008), 'Real-time capable system for hand gesture recognition using hidden markov models in stereo color image sequence real-time capable system for hand gesture recognition using hidden markov models in stereo color image sequences'.
URL: *https://www.researchgate.net/publication/228363421*

ft.lk (2019).
URL: *https://www.ft.lk/Other-Sectors/Pizza-Hut-embraces-inclusivity/57-679358*

Haputhanthri, H. H. S. N., Tennakoon, H. M. N., Wijesekara, M. A. S. M., Pushpananda, B. H. R. & Thilini, H. N. D. (2022), 'Multi-modal deep learning approach to improve sentence level sinhala sign language recognitio', *International Journal on Advances in ICT for Emerging Region* **259**.

He, K., Zhang, X., Ren, S. & Sun, J. (2015), 'Deep residual learning for image recognition'.
URL: *https://arxiv.org/abs/1512.03385*

Holmes, R., Rushe, E., Coster, M. D., Bonnaerens, M., Satoh, S. . I., Sugimoto, A. & Ventresque, A. (2023), 'From scarcity to understanding: Transfer learning for the extremely low resource irish sign language'.
URL: *https://wfdeaf.org/our-work/*

Hu, L., Gao, L., Liu, Z. & Feng, W. (2023), 'Continuous sign language recognition with correlation network'.
URL: *http://arxiv.org/abs/2303.03202*

Kumar, R. & Bajpai, A. (2023), 'Mediapipe and cnns for real-time asl gesture recognition'.

Latif, G., Mohammad, N., Alghazo, J., AlKhalaf, R. & AlKhalaf, R. (2019), 'Arasl: Arabic alphabets sign language dataset', *Data in Brief* **23**.

Li, D., Opazo, C. R., Yu, X. & Li, H. (2020), 'Word-level deep sign language recognition from video: A new large-scale dataset and methods comparison'.
URL: *https://dxli94.github.io/*

Liyanaarachchi, P., Dinithi, R. & Shakya, N. (2021), 'Signing dataset for the sinhala sign language'.
URL: *https://www.researchgate.net/publication/353598906*

Madushanka, A., Senevirathne, R., Wijesekara, L., Arunatilake, S. & Sandaruwan, K. (2016), 'Framework for sinhala sign language recognition and translation using a wearable armband'.

MediaPipe Solutions guide (2024). Accessed: 8, 2024.
URL: *https://ai.google.dev/edge/mediapipe/solutions/guide*

Microsoft (2018), 'Ms-asl'. Accessed: 8, 2024.
URL: *https://www.microsoft.com/en-us/research/project/ms-asl/*

Moryossef, A., Tsochantaridis, I., Dinn, J., Camgöz, N. C., Bowden, R., Jiang, T., Rios, A., Müller, M. & Ebling, S. (2021), 'Evaluating the immediate applicability of pose estimation for sign language recognition'.

Pramanto, H. & Suharjito, S. (2023), 'Continuous sign language recognition using combination of two stream 3dcnn and subunet', *JURNAL TEKNIK INFORMATIKA* **16**, 170–184.

Quesada, L., López, G. & Guerrero, L. (2017), 'Automatic recognition of the american sign language fingerspelling alphabet to assist people living with speech or hearing impairments', *Journal of Ambient Intelligence and Humanized Computing* **8**, 625–635.

Selvaraj, P., NC, G., Kumar, P. & Khapra, M. (2021), 'Openhands: Making sign language recognition accessible with pose-based pretrained models across languages'.
URL: *http://arxiv.org/abs/2110.05877*

Sridhar, A., Ganesan, R. G., Kumar, P. & Khapra, M. (2020), Include: A large scale dataset for indian sign language recognition, Association for Computing Machinery, Inc, pp. 1366–1375.

tensorflow (2025).
URL: *https://www.tensorflow.org/*

Tur, A. O. & Keles, H. Y. (2021), 'Evaluation of hidden markov models using deep cnn features in isolated sign recognition', *Multimedia Tools and Applications* **80**, 19137–19155.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A., Kaiser, . & Polosukhin, I. (2017), Attention is all you need, *in* 'Advances in Neural Information Processing Systems', Vol. 30, pp. 5,6.

VGT, C. (2022), 'Corpus vlaamse gebarentaal (corpus vgt)'. Accessed: 8, 2024.
URL: *https://taalmaterialen.ivdnt.org/download/corpusvgt1-0/*

WLASL (World Level American Sign Language) Video (2020). Accessed: 8, 2024.
URL: *https://www.kaggle.com/datasets/risangbaskoro/wlasl-processed*

Wong, S.-F. & Cipolla, R. (2005), 'Lncs 3766 - real-time adaptive hand motion recognition using a sparse bayesian classifier'.

A Additional Resources

All code, training logs, extended visualizations, and result plots are available at the following links:

- **GitHub repository:** `https://github.com/Neethamadhu-Madurasinghe/sign_language`
- **Full training plots, confusion matrices and trained models (Drive):** `https://drive.google.com/drive/folders/15u__Udm1fbYDjeFtY9LSoEGdUVOyDEWz?usp=drive_link`
- **Google sheet containing collected experiment results:** `https://docs.google.com/spreadsheets/d/1iKVuq3n6J1Jo4dJyccBP5rSivK9pVD7f1DIYQEhYpI4/edit?usp=sharing`

These resources include:

- Complete model training scripts and configuration files
- Training/validation loss and accuracy plots for all runs
- Raw evaluation metrics (accuracy, F1-score, etc.) for each experiment
- Pre-processing code and data analytics
- All the plots that included in drive link
- .h5 files for trained models