Sign Language Recognition with 3DCNN and SlowFast Neural Network Models

by

Mar Galiana Fernández

Supervised by

Dr Ahir Bhalerao

Department of Computer Science

University of Warwick

6th September 2023

Keywords: Sign Language Recognition, SlowFast Neural Network, 3DCNN, Data Processing Techniques, Face Recognition, MediaPipe, WLASL Dataset, MSASL Dataset.

Abstract

This research project delves into the realm of Sign Language Recognition (SLR) employing advanced models, such as, the 3DCNN and the SlowFast Neural Network, with the overarching goal of improving accuracy. The study encompasses rigorous experimentation, focusing on data processing techniques and model optimisation. Under computational constraints, 3DCNN achieved an accuracy of 68. 25%, and SlowFast neural network reached 66. 23%. The project emphasises the significance of data-processing methods, showcasing the importance of features such as facial expressions and hand movements in sign interpretation. Additionally, a manually curated set of 50 labels, representing various aspects of sign language, was used for model training. This report not only highlights achievements within limitations but also lays the groundwork for future endeavours.

Acknowledgements

I would like to express my sincere gratitude to my supervisor, Dr. Ahir Bhalerao, for his exceptional guidance and mentorship throughout this endeavour. His support and valuable feedback have been essential in shaping our work. I also want to extend my heartfelt thanks to my friends and family for their unwavering moral support when it was needed the most. In particular, I deeply appreciate my sister, Laura, and my parents, Jorge and Mª Carmen, for granting me the opportunity to study abroad and gain invaluable experiences that have significantly contributed to my personal and academic growth.

Contents

List of Figures

List of Tables

1 Introduction

This chapter gives an overview of the techniques used for Sign Language Recognition. It describes the aim of the thesis and the motivation behind it.

1.1 Overview

Sign language serves as a vital communication tool for individuals with speech or hearing impairments. However, it's important to note that there is no single universal sign language; instead, there are approximately 140 distinct sign languages according to Ethnologue [2]. Each country often possesses its own national sign language and finger-spelling alphabet. Consequently, there arises a critical need for systems capable of recognising sign gestures and effectively conveying their meaning to individuals who may not have knowledge of these languages. These systems are collectively referred to as Sign Language Recognition (SLR).

The objective of this project is to conduct a comprehensive evaluation of the current state-of-the-art in SLR. By doing so, this project aims to identify any existing gaps or shortcomings in the field, paving the way for the development and implementation of more advanced SLR systems.

1.2 Motivation

Sign languages, unfortunately, are not typically included in mainstream education, resulting in only a small fraction of the global population having proficiency in these languages. This limitation poses significant challenges for individuals with speech or hearing impairments, especially when attempting to convey essential messages or carry out everyday tasks. While there is a growing awareness of this issue, society's current level of knowledge about sign languages remains insufficient to create an inclusive environment for people living with these disabilities [3, 4].

The primary objective of this research project is to expedite the process of raising awareness and understanding of sign languages. By doing so, the project aims to accelerate progress toward building a more inclusive society that accommodates and supports individuals with speech and hearing impairments.

1.3 Aim of the thesis

This thesis aims to develop a robust sign language recognition system based on American Sign Language (ASL). American Sign Language was chosen due to its abundance of available data. The primary input for this system will be video recordings of sign language interpreters, with corresponding text outputs for each sign.

The system utilises video as the input data source because of its accessibility. Video data can be captured using readily available and cost-effective camera devices. Unlike many existing systems that rely on expensive and complex sensor technology, the goal is to streamline and simplify the user experience, making sign language communication more accessible to individuals who cannot rely on verbal language.

Throughout the project, a comprehensive study of existing Sign Language Recognition (SLR) systems will be conducted, this will involve assessing their suitability based on specific requirements. Various datasets, data preprocessing techniques, different model architectures, and optimisation strategies will be explored. The objective is to propose an enhanced SLR system that can address the unique challenges posed by sign language recognition.

The structure of the report is as follows:

- Section 2 provides an overview of previous research in the field, highlighting significant advancements.
- Section 3 serves as the core of this report, offering detailed insights into the experiments. Each experiment's initial goals, underlying intuition, expectations, actual outcomes, and discussions on goal achievement or alternatives are presented.
- Section 4 consolidates and summarises the results obtained from the experiments.
- Subsequent sections cover project management, limitations, and ethical considerations associated with the project.
- The report concludes with suggestions for future work (Section 8) and a comprehensive summary (Section 9). An appendix containing relevant figures referenced throughout the report is provided.

2 Background and research

Various techniques have been used in Sign Language Recognition (SLR) based on system requirements. Wadhawan and Kumar [1] summarised the rates of different papers on American Sign Language (ASL) training models. They compared the output rates based on five characteristics:

- Data Acquisition: Different methods were used to acquire data, including cameras and sensors. Cameras provided either images or videos, while sensors included gloves, Kinect, arm sensors, electroencephalogram, and leap motion.
- Type of Signs: Signs can be static (no movement, e.g., ASL alphabet) or dynamic (requiring movement for interpretation). Static signs often used camera-acquired images, while dynamic signs used videos.
- Modelling Algorithm: Various algorithms and techniques were employed. Previous papers using cameras and dynamic signs used Dynamic Time Warping (DTW) and Support Vector Machine (SVM).

Wadhawan and Kumar [1] also discuss other aspects regarding the dataset used to train the model. The main characteristics are:

- Interpreter mode. Comparing whether it is either isolated, continuous or both. Isolated signing refers to independent signs without any connections to preceding or succeeding signs. Continuous signing involves several signs without distinct pauses between them.
- Number of hands needed to interpret the signs. It can be single- or double-handed.

Table 1 summarises the above information. It is taken from the article by Wadhawan and Kumar [1], although only the information relevant to this article has been retained. Only the articles training the model with data containing dynamic signs or both (dynamic and static) are presented in the table.

Paper	**Data Acquisition**	**Gestures**	**Technique**	**Rate**
Oz and leu [5]	Gloves	Dynamic	NN	95%
Sun et al [6]	Kinect	Dynamic	Latent SVM	86%
Sun et al. [7]	Kinect	Dynamic	Adaboost	86.8%
Jangyodsuk et al. [8]	Camera	Both	DTW	93.38%
	Kinect	Both	DTW	92.54%
Wu et al. [9]	Arm sensors	Dynamic	Decision tree	81.88%
			SVM	99.09%
			NN	98.56%
			Naïve Bayes	84.11%
Usachokcharoen et al. [10]	Kinect	Dynamic	SVM	95%
Savur and Sahin [11]	Arm band	Both	SVM	82.3%
Sun et al. [12]	Kinect	Both	Latent SVM	86%
Kumar et al. [13, 14]	Camera	Static	SVM	93%
		Dynamic	SVM	100%
Savur and Sahin [15]	Armband	Dynamic	SVM and ensemble learner	60.85%
AlQattan and Sepulveda [16]	Electroence-phalogram	Dynamic	LDA	75%
			SVM	76%

Table 1: Summarised review of ASL recognition systems comparing the Reported Rate obtained. This table is located in the Sign Language Recognition Systems: A Decade Systematic Literature Review article [1].

From table 1, the most remarkable papers for this report are the ones from Jangyodsuk et al. [8] and Kumar et al. [13, 14]. Both researchers have used cameras as the data acquisition method, and the data set used to train the model uses dynamic signs.

Kumar et al. [13, 14] developed a sign language recognition system for recognising both static and dynamic signs in American Sign Language. They focused on predicting the letters a-z (where only the letters j and z are dynamic). The system was able to perform dynamic backgrounds with minimal decorations, as it relies on skin colour segmentation to identify gestures. Since the signs they wanted to predict do not use a facial expression, they removed this section of the video using Viola-Jones face detection followed by subtraction of the detected region. Once the data was processed, they extracted a curved feature vector, following previous work from Bhuyan et al.,

[17]. Afterwards, these feature vectors were classified using pre-trained SVM classifiers. Static and dynamic gestures were differentiated by measuring the distance travelled by the hand in subsequent frames. Dynamic gesture recognition was performed using four gestures for testing, which are: "j", "z", "no", and "goodbye", achieving an accuracy of 100%.

Jangyodsuk et al. [8] employed a dataset consisting of videos taken with a standard RGB camera, with three signers each making 1,113 signs, for a total of 3,339 different signs. They followed previous work by Dalal et al. [18], who applied the Dynamic Time Warping (DTW) method to compare the hand trajectory using a Histogram of Oriented Gradient (HoG) to represent hand shape. However, they standardised the features to have a mean value of 0 and a standard deviation of 1. With this improvement, their accuracy increased, on average, by about 10%. Using Hand trajectory matching with hand shape distance using HoG features as shape representation, they archived an accuracy of 93.38

There exist other papers published after the Wadhawan and Kumar article [1] was released which have other interesting methods applied.

Borg and Camilleri [19] implemented, in 2020, a two-stage system, which has the hand key points features obtained via OpenPose as the input to the model. They use Hidden Markov models (HMMs) to obtain the subunits of sign concatenation (SU). The SU descriptors are employed to train the SU Recurrent Neural Networks (RNNs), using a Connectionist Temporal Classification (CTC) framework to handle the temporal sequence. Once the SU RNNs are trained, a second-level RNN is added for sign recognition. RWTH-Phoenix Weather [20] was the dataset used which contained 1230 unique signs with 9 signers. With this implementation, they archived an accuracy of 71.9%.

Zheng et al. [21] proposed a model that reduced the training size data by 9.3%, compared with the state-of-the-art of 2020. This model used the Frame Stream Density Compression (FSDC) algorithm to detect and reduce redundant similar frames, which shortens long sign sentences without losing information. They further implemented a temporal convolution (T-Conv) connected to a dynamic hierarchical bidirectional GRU (DH-BiGRU). The RWTH-Phoenix Weather Dataset [20] was used, which is the same as the one utilised by Borg and Camilleri [19].

Al-Hammadi et al. [22] used a 3D Convolutional Neural Network (3DCNN)

where three instances of the 3DCNN structure were trained to extract the hand gesture features from the beginning, middle, and end of the video sample. Afterwards, they studied three techniques for feature fusion: multilayer perceptron (MLP) neural network, long short-term memory (LSTM) network, and stacked autoencoder. Using the 3DCNN and MLP in a 40 classes dataset called KSU-SSL dataset, they reached a recognition rate of 84.38%.

Finally, Hassan et al. [23] conducted an experiment using various models, with the most successful one being the SlowFast Neural Network. They utilized a pre-trained model called *SLOWFAST_8×8_R50* obtained from the PySlowFast GitHub repository [24]. The experiment utilized the WLASL [25] dataset and achieved predictions for 300 different labels, resulting in a TOP 1 accuracy of 79.34%, which implied an improvement of 23.2% over the previous state-of-the-art performance. The researchers mentioned that the limitations they encountered were related to the time-consuming nature of model training, which required a total of twenty-four hours spread across multiple days, as well as some hardware limitations.

Even though there is intensive research conducted on gesture recognition, the majority of them have been executed using data coming from complex hardware and inconvenient for the user to carry on a daily basis. Research so far indicates that the architectures being used are vastly differing, therefore it is necessary to bring all the information together and try to get the best out of each one of them.

3 Experiments

3.1 Experiment 1: First model implementation

The objective of this experiment was to develop an initial model as a preliminary attempt, aimed at understanding the complexities of the problem being addressed.

As the primary focus of this experiment was not to achieve maximum performance, the chosen model configuration was not optimised. The number of epochs and batch size were intentionally kept at their minimum values to accelerate the results without enduring an extended training duration.

3.1.1 Data selection

To begin with, it was essential to select an appropriate dataset that could be utilised in future experiments. Additionally, another significant choice to make was the proportion of training, testing, and validation data, which would be contingent on the number of videos obtained from each label.

Numerous datasets were evaluated, all sourced from other papers that had also implemented an SLR using the ASL. This choice was deliberate as it provides a more accurate basis for comparing the performance of other models with that presented in this report under similar conditions.

The dataset needed to contain labelled videos, as the data acquisition and labelling are beyond the scope of this project. The following datasets have been examined in order to determine which was going to be the most accurate to the project requirements.

- Word-Level American Sign Language (WLASL) [25]
- MS-ASL dataset [26]
- The American Sign Language Lexicon Video Dataset (ASLLVD) [27].
- Datasets from the Kaggle [1] community.

[1] https://www.kaggle.com

The ASLLVD was gathered at Boston University and encompassed more than 3,000 signs produced by 1-6 native ASL signers. Sadly, this dataset had to be excluded due to the requirement of submitting a petition for download, which was never approved. Among the options available on Kaggle, the sign count in each dataset was considerably lower and less reliable compared to other datasets.

The samples within the MS-ASL dataset were collected by Microsoft, featuring 1,000 distinct labels and involving over 200 signers. Furthermore, the WLASL dataset offers a wider range of backgrounds, speaker speeds, physiques, and camera orientations. Additionally, it is the largest ASL dataset available, featuring 2,000 common words in ASL. Both the MS-ASL and the WLASL datasets were viable options, but eventually, the WLASL was chosen, as it was the most commonly used dataset, which helps compare the results of the proposed model with the results from other papers.

Once the dataset was downloaded, 10 random labels were chosen: drink, trade, before, bowling, computer, cool, go, thin, help and tall. There were 12 samples for each label, leaving 8 for training and 2 for testing and validation purposes. Although this number of samples may not be sufficient to train and validate a robust model, it is important to note that the purpose of this experiment is to train a model for the first time, understand the data processing involved, and anticipate the type of results that will be obtained. Specifically, 70% of the samples were allocated for training, while 20% were used for validation and the remaining 20% for testing.

3.1.2 Data processing

In the first experiment, a series of data processing tests were conducted to determine the relevant portion of the sign interpretation within the video. For each video, 20 consecutive frames were extracted from four different starting points. These frames were obtained from the beginning, middle, and end of the video, as well as randomly selected positions. This approach helped reduce the data size by 10 frames per video while capturing different sections of the sign interpretation.

3.1.3 Model selection

After an in-depth exploration of the state-of-the-art in section 2, a range of promising techniques came under evaluation. These encompassed:

- Support Vector Machine [13, 14].
- Dynamic Time Warping Using a Histogram of the Orientated Gradient [8].
- Hidden Markov models combined with Recurrent Neural Network [19].
- Temporal convolution network connected to a dynamic hierarchical bi-directional GRU. [21].
- 3D Convolutional Neural Network using a multilayer perceptron for feature fusion. [22].
- Slow Fast Neural Network [23].

Based on the outcomes derived from these techniques, the 3DCNN model emerged as the preferred choice for the initial experiment. Specifically, the selected model was the pre-trained MoViNet-A0-Base model sourced from TensorFlow [2]. This particular model had undergone training on the kinetics dataset, achieving a TOP 1 accuracy of 72.28% while operating with an input size of 50x172x172 pixels.

In this context, the model was re-trained using the 10 distinct labels mentioned in section 3.1.1. This retraining involved using a batch size of 8 over the course of 10 epochs, using 20 frames from each video in the training process. Each label was assigned 12 samples for retraining.

3.1.4 Results

The outcomes of each data processing test are meticulously documented in table 2.

A notable observation is the 20% decrease in model accuracy when utilizing the final frames of the video. This outcome implies that the pivotal segment of the video lies within both the initial and middle sections, whereas

[2]https://github.com/Atze00/MoViNet-pytorch

Test	Frames position	Accuracy
1	Beginning	40%
2	Middle	40%
3	End	20%
4	Random Start	40%

Table 2: Results of the second experiment after optimising the model

the end of the video proves less informative. In particular, even when starting from random positions, the accuracy remains consistent at 40%, confirming the importance of both the beginning and middle sections to capture essential information for accurate interpretation.

The results findings of this experiment are as expected due to the fact that at the last moments of each video, the signers often await the recording's conclusion without contributing any valuable sign content.

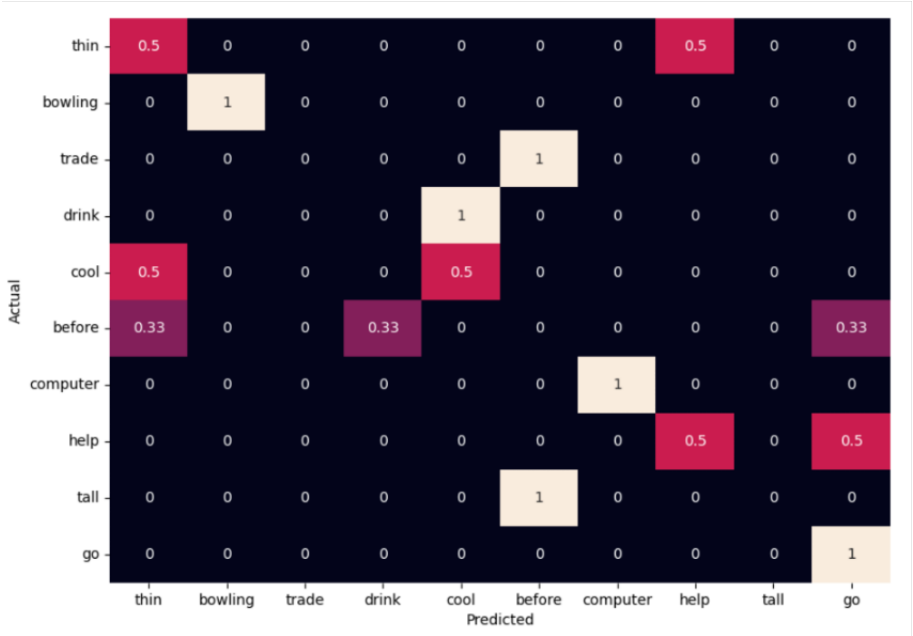

Figure 1: The confusion matrix when testing the 3DCNN model in the first experiment.

Looking at the results from the confusion matrix in figure 1, it's noticeable that the labels "Trade," "Tall," and "Before" consistently get mixed up. After closely examining these labels, it's clear that they all involve signs performed in the same space using both hands and without much movement

in other parts of the body. This similarity in how these signs are executed is likely causing the model to confuse them with each other.

3.1.5 Results discussion

The main objective of the initial experiment was to develop an early model, aiming to gain insights into the complexities of the addressed problem.

Ultimately, this goal was successfully achieved. A thorough analysis of various available datasets took place, leading to a better understanding of the specific requirements and constraints of the problem, including considerations such as the labels the model could effectively learn. During data preprocessing, it became evident that information in the samples was often irrelevant, especially towards the end of the video sequences. In future experiments, this insight will inform data extraction, now encompassing frames from both the initial and middle segments of the samples.

Model selection provided valuable insights into widely-used models and guidelines on obtaining pre-trained models. However, a notable challenge emerged in adapting the input data to align with the shape and size prerequisites of the chosen pre-trained model. This challenge arose due to the less-than-ideal documentation accompanying pre-trained models and the unique transformation requirements for each dataset.

In summary, the achieved accuracy, considering this as the inaugural trial, proved commendable. Several labels, including "bowling," "cool," "computer," and "go," were consistently predicted accurately, as corroborated by the confusion matrix in Figure 1. While numerous areas for improvement have been identified, the overarching goal of this experiment has been accomplished.

3.2 Experiment 2: Compare the SlowFast neural network to 3DCNN

The aim of the second experiment was to introduce the SlowFast Neural Network and compare its performance with that of the 3DCNN, specifically by

focusing on the body areas of the videos that hold the sign information.

The intuition behind this experiment stems from the inherent model architecture of the SlowFast Neural Network. This architecture is believed to offer advantages in the realm of sign language recognition, as discussed in Section 1.3. Moreover, the experiment hypothesises that by concentrating on the primary movement areas within signs, performance enhancements can be achieved.

3.2.1 Data selection

For this experiment, the number of labels was increased by 10, resulting in a total of 20 labels. These labels were randomly selected and included words such as book, drink, computer, before, chair, go, clothes, who, candy, cousin, dead, fine, no, thin, walk, year, yes, all, black, and help.

Additionally, the number of samples per video was elevated. Models were trained using 22 videos for each label, all obtained from the WLASL dataset. The data allocation consisted of 70% for training, 15% for testing, and the remaining 15% for validation.

3.2.2 Data processing

As discussed in Section 1, signs are distinguishable by factors such as hand gestures, facial expressions, and body movements. Building upon this understanding, the videos were processed to highlight these specific aspects of the signer's body, with a focus on detecting hands and the face in each frame of every sample.

Each video frame underwent additional processing to test four different improvements and determine which ones enhanced the models' performance. These enhancements were selected based on the body parts visible in the original image. The various types of enhancements can be seen in Figures 2, 4, 3, and 5. These figures illustrate the four distinct ways in which the images were altered. In each image, a person is shown using sign language. On the left, the unmodified image is visible, while on the right, the adjusted version designed to facilitate model comprehension is presented.

The first type, referred to as "ALL" during the experiment, is depicted in Figure 2. This image shows a person using sign language to convey the word "secretary." The modified image is divided into four sections: the upper-left part focusses on the left hand, the upper-right part accentuates the right hand, the lower-left part highlights the face, and the lower-right part displays the entire image with a subtle blur. The blur is applied to help highlight additional details while retaining the main context.

Figure 2: Visual representation of video processing in Experiment 2. The image on the left presents the untouched original, while the image on the right showcases the result after employing the "ALL" processing approach.

In the second type, which will be named "FACE AND HANDS" during the execution of the experiment, the input frames are transformed to the model as shown in the example illustrated in Figure 3. The illustration portrays an interpreter performing the ASL sign for the word "angry". This processed image is divided into three sections: the top left zooms in on the left hand, the top right emphasises the right hand, an the total weight of the bottom highlights the face. The intuition behind this experiment is that the significant part of the sign interpretation relies on the hands gesture and face expression, the rest of the body is static, ergo it is irrelevant information that the model does not need to understand the sign.

The third category, denoted as "BODY AND HANDS" throughout the experiment, involves transforming the input frames for the model, as exemplified in the illustration of Figure 4. This depiction features an interpreter

Figure 3: Visual representation of video processing in Experiment 2. The image on the left presents the untouched original, while the image on the right showcases the result after employing the "FACE AND HANDS" processing approach.

enacting the ASL sign for the word "book." The altered image shares similarities with the "FACE AND HANDS" processing approach. However, in this instance, the entirety of the bottom section shows the complete image with a subtle blur. The rationale behind this processing experiment is that although certain signs require a facial expression to be represented accurately, these instances are relatively infrequent. Most often, the uniqueness of a sign resides in the hand movements themselves. As such, the face need not be prominently highlighted; including the entire image in the lower part of the frame provides sufficient context for interpreting the sign. The introduction of the blur serves to emphasise other relevant aspects while preserving the overall context.

The final processing technique, referred to as "HANDS" throughout the experiment, involves transforming the input frames for the model. This technique is illustrated in Figure 5, where an interpreter is depicted performing the ASL sign for the word "help." The adjusted frame comprises two images: one showcasing the left hand and the other displaying the right hand. Both the face and the original image have been omitted from this version. This approach is based on the idea behind the "BODY AND HANDS" image processing technique, which aims to reduce the focus on the face. Furthermore, the exclusion of the original image is based on the understanding that the most important information in sign language is contained within the hand movements. By emphasising the hands over other visual elements, the

Figure 4: Visual representation of video processing in Experiment 2. The image on the left presents the untouched original, while the image on the right showcases the result after employing the "BODY AND HANDS" processing approach.

model's attention is narrowed, thus improving its ability to recognise and accurately interpret the complex gestures of sign language.

To detect hands in the original images, a variety of libraries were used. Initially, OpenPose was used, a choice also made by Borg and Camilleri [19]. However, OpenPose's outcomes were rather imprecise, prompting the exploration of alternative libraries. Ultimately, Mediapipe was chosen for hand detection and differentiation between the left and right hands. Although slight errors may still exist in certain frames, Mediapipe showed considerable improvement compared to OpenPose.

For face detection, the library "face_recognition," developed using advanced techniques from dlib, was employed. This library consistently delivers accurate results in face-detecting tasks.

The completed implementation for this stage of the experiment is available within the script titled "processing/process_videos.py." It can be found in the specified GitHub repository in Section 1.

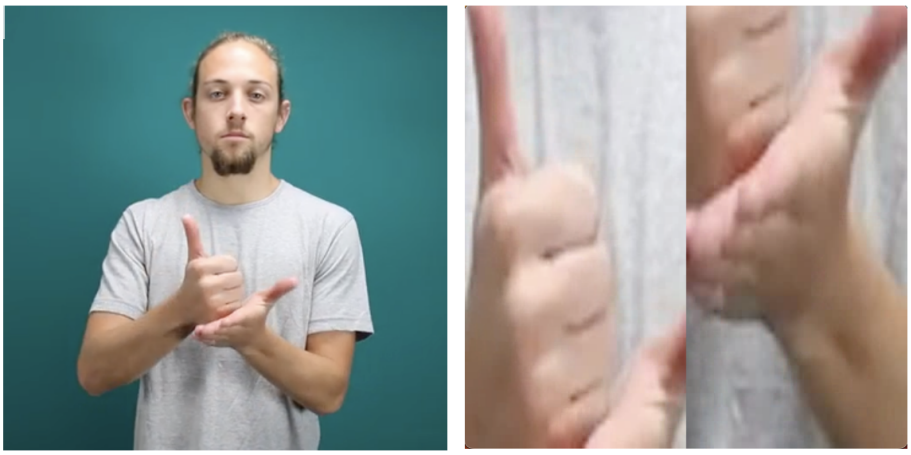

Figure 5: isual representation of video processing in Experiment 2. The image on the left presents the untouched original, while the image on the right showcases the result after employing the "HANDS" processing approach.

3.2.3 Model selection

Two models were employed for training within this experiment: the SlowFast Neural Network and the 3DCNN. Both models had previously been pre-training, which helped reduce the computational requirements when retraining them with the WLASL dataset.

The SlowFast Neural Network, created and trained by Facebook [28], was trained using the Kinetics-400 [29], Kinetics-600 [30], Charades [31], and AVA [32] datasets. Damen et. al. [24] achieved a TOP 1 accuracy of 77% using this model with the same dataset employed in this experiment.

For the 3DCNN model used in the initial experiment, its selection was based on its simplicity and efficiency in terms of memory and computational time. However, as the objective of this experiment was to attain reliable performance, a different pre-trained model was necessary. The chosen model is known as R3D18 and was obtained from Torchvision [33]. This model had been pre-trained for action recognition purposes and achieved an accuracy of 74.3% using the Kinetics dataset [34].

3.2.4 Results

The precision obtained training the 3DCNN and the SlowFast Neural Network using the four types of processing techniques mentioned in Section 3.2.2 is summarised in Table 3.

Processing data type	**3DCNN**	**SlowFast**
All	14.58%	58.33%
Face and Hands	14.58%	62.50%
Body and Hands	18.37%	**63,27%**
Hands	**26,53%**	53.06%

Table 3: A comparison of accuracy performance between the 3DCNN and SlowFast Neural Network models, considering four different types of data processing.

For the ALL processing data type, the SlowFast Neural Network achieved an accuracy of 58.33%, as depicted in Figure 13 for the loss function and Figure 12 for the confusion matrix in Section A.1. Using the 3DCNN approach, an accuracy of 14. 58% was achieved, with its loss function shown in Figure 15 and the corresponding confusion matrix in Figure 14, both available in Section A.2.

For your convenience, the code used to train the 3DCNN can be accessed through the following code repository: GitHub link. To replicate the SlowFast experiment, the relevant code can be found within this experiment's code repository: GitHub link.

Similarly, for the FACE AND HAND processing data type, the SlowFast Neural Network achieved an accuracy of 62.50% as shown in Figure 21 for the loss function and Figure 20 for the confusion matrix in Section A.5. Utilizing the 3DCNN approach yielded an accuracy of 14.58% with its loss function illustrated in Figure 23 and the corresponding confusion matrix in Figure 22, both available in Section A.6. The relevant code for this experiment can be found in the code repository GitHub link.

In the case of the BODY AND HAND processing data type, the SlowFast Neural Network achieved an accuracy of 63.27% as highlighted in Figure 17 for the loss function and Figure 16 for the confusion matrix in Section A.3. The 3DCNN approach resulted in an accuracy of 18.37% with its loss func-

tion shown in Figure 19 and the corresponding confusion matrix in Figure 18, both available in Section A.4. The relevant code for this experiment can be found in the following code repository GitHub link.

Lastly, employing the HANDS processing data type led to a SlowFast neural network accuracy of 53. 06%, with the loss function displayed in Figure 25 and the confusion matrix in Figure 24 within Section A.7. Using the 3DCNN approach resulted in an accuracy of 26.53%, with its loss function presented in Figure 27 and the corresponding confusion matrix in Figure 26, both available in Section A.8. The relevant code for this experiment can be found in the code repository GitHub link.

Analysing the results obtained, it can be observed how the SlowFast Neural Network gives a higher performance in all data types processed, archiving a top accuracy of 63.27% when using the body and hands processing type. And taking a look at the 3DCNN, it seems like it is not even learning from the training, given the low accuracy that it is returning. The 3DCNN is a commonly used approach for video recognition, as mentioned in the state-of-the-art section 2, which implies that it should have a higher accuracy than the one obtained from this experiment. The following experiments will be focused on understanding what is happening to the model and try to improve it.

Upon analysing the results obtained, it becomes evident that the SlowFast Neural Network consistently outperforms the 3DCNN across all processing data types. In particular, the SlowFast neural network achieved a remarkable accuracy peak of 63.27% when applied to the body and hands processing type.

However, a closer examination of the 3DCNN's performance reveals a concerning trend: it appears to struggle in acquiring meaningful insights from the training data, leading to a noticeably low accuracy rate. It can be verified evaluating the loss function and confusion matrix of the four different data processing techniques when using the 3DCNN model. Given the significance of 3DCNN as a widely used technique for video recognition, as mentioned in Section 2, the disparity between its anticipated performance and the observed results is concerning.

The following experiments will be directed towards uncovering the factors contributing to this underperformance and formulating strategies to enhance the 3DCNN's efficacy. This proactive investigation is crucial for gaining in-

sights into the model's limitations and, ultimately, for steering improvements in its functionality.

3.2.5 Results discussion

The objective of the second experiment was to explore the most effective data processing approach by comparing the SlowFast Neural Network with the 3DCNN models. The underlying hypothesis was that the SlowFast Neural Network, with its unique architecture, might outperform the 3DCNN when applied to the specific data characteristics of the problem.

Expanding the number of labels and sample sizes led to more precise results. Various data processing methods were successfully applied, enabling the detection of specific aspects within interpreters with high precision. The challenge arose during model selection as a deep understanding of the SlowFast architecture was required to correctly transform the input data into the two channels necessary for proper input.

However, in the end, the obtained results did not align with the initial expectations of this experiment. Due to the 3DCNN's limited ability to learn effectively, a direct comparison between these two models could not be executed. The 3DCNN failed to adequately showcase the model's potential. Nonetheless, when focusing solely on the SlowFast model, it was possible to identify the data processing approach that yielded the highest performance.

In future experiments, the emphasis will be on enhancing the training process of the 3DCNN to complete the current experiment's objective. This will help determine whether the initial intuition regarding the SlowFast Neural Network's superior performance holds true or not.

3.3 Experiment 3: Join Datasets

The primary objective of the third experiment was to enhance the performance of the 3DCNN model. The rationale behind this initiative stems from the observation that the model's learning capacity was potentially limited because of the restricted number of available samples. As detailed in the analysis mentioned in Section 3.1.1, a preliminary study was conducted on

the existing ASL dataset.

The proposed solution was to augment the existing dataset with a new one. This augmentation aimed not only to increase the dataset's overall sample size but also to facilitate improved generalisation by providing the model with a more diverse range of examples to learn from.

Given the use of two different datasets, this experiment also explored how combining them can enhance the model's performance. This evaluation involved deciding whether one dataset should be primarily used for training, or if both should be divided and used for complementary purposes.

3.3.1 Experiments dataset requirement

The primary dataset used in this experiment remained consistent with the dataset employed in previous experiments, named the WLASL dataset [25]. MSASL [26] was the secondary dataset. As detailed in Section 3.1.1, the choice of this dataset was identified as the most suitable option.

More labels were added to train the model, with a total of 50 labels. Given the necessity of downloading data, it was logical to do so collectively. However, this time the labels were manually selected, taking into account the type of gesture and the significance of the word. Priority was given to words that require early acquisition.

Upcoming experiments will delve into how various gestures impact data processing. Therefore, for this experiment, it was ensured that three types of signs were included: face motions (expressing emotions such as anger or happiness), body motions (involving hands, shoulders and arms, like the sign for "big"), and hand gestures (primarily involving hand movement). A significant proportion of signs involved hand movement, encompassing 68% of the selected signs, while face and body gestures each constituted 16%. Specifics can be found in Table 4.

For the execution of this experiment, acquiring the new dataset required certain steps. This process demanded increased computational memory and time. Although downloading around 22 videos for each of the 50 labels in the WLASL dataset took approximately 2 hours, the new dataset required approximately 10 hours. This increase in download duration can be

Gesture Type	Included Labels
Face Motion	Laugh, cry, angry, surprise, happy, think and delicious.
Body Motion	Dance, fly, hello, teacher, swim, important, crazy and swing.
Hand Motion	Sister, bird, book, friend, doctor, eat, nice, yes, learn, no, like, want, deaf, school, finish, white, fish, sad, table, father, milk, brother, paper, mother, water, help, yellow, hungry, drink, careful, coffee, phone and more.

Table 4: Labels categorised by gesture type in the 50-label dataset.

attributed to two primary factors. Firstly, the MSASL dataset contained double the number of videos compared to the WLASL dataset. Secondly, the new dataset's videos were notably longer in duration, often extending up to five minutes, in contrast to the typical three-second duration of regular sign videos.

A distinctive feature of the videos in the new dataset was their extended content, incorporating multiple signs within a single video. This inherent characteristic resulted in the prolonged duration of the video. The challenge lay in the fact that the downloaded videos needed subsequent editing to retain only the pertinent sections.

Throughout this experiment, the ALL data processing technique was applied consistently. The plan was to evaluate other methods once the model demonstrates successful learning.

3.3.2 Using WLASL dataset for Training

While the primary goal of this experiment was to enhance the performance of the 3DCNN model, it will also be conducted on the SlowFast neural network to assess whether increasing the sample size could lead to improvements. The resulting performance of each model, now that the sample size has been doubled, is summarised in Table 5, which presents the outcomes of this experiment.

For the 3DCNN, the confusion matrix and loss function can be found in Figures 30 and 31 respectively, detailed in Section B.2. These visualisations

Training Dataset	Validation/Test Dataset	3DCNN	SlowFast
WLASL	WLASL	**14.58%**	**58.33%**
WLASL	MSASL	3.60%	44.44%

Table 5: Accuracy achieved by the 3DCNN and SlowFast models, trained using the WLASL dataset and evaluated on the MSASL dataset. Both datasets were previously processed using the ALL data processing type.

highlight that the model is still struggling to learn, as evidenced by the accuracy dropping to 3.60%. On the other hand, the SlowFast model exhibits learning capabilities, as indicated by the confusion matrix in Figure 28 and the loss function in Figure 29, both available in Section B.1. However, the precision has decreased from 58.33% to 44.44%. The code for this experiment can be found in the repository GitHub link.

The reduction in accuracy in both models, despite the augmented sample size, may be attributed to the substantial disparities between the MSASL and WLASL datasets. This intuition will be further explored in upcoming experiments.

3.3.3 Increase batch size using mixed precision training

As increasing the sample size did not yield significant improvements in the 3DCNN model's performance, another potential issue could be the batch size used during the training process. In previous experiments, the batch size was set to 5. Attempting to increase it to 16 resulted in a CUDA out of memory error. Upon further investigation, it was discovered that employing mixed precision training, which reduces the number of decimal places in the neural network's weights to conserve memory, might be a viable solution.

Implementing mixed precision training implies that the model may not attain its peak performance, but raises the possibility that, when combined with an increased batch size, it could outperform the existing small batch size configuration.

Shifting to CPU-based training rather than utilising the GPU was not a viable alternative. Such an approach would have considerably prolonged the training time, imposing substantial constraints on the project's scope and

feasibility. Unfortunately, the CUDA capacity was only capable of training the models using a batch size of 10, any higher number would result in an out of memory error.

Table 6 provides a summary of the accuracy achieved in the various trial runs of the current experiment. Notably, the accuracy of the 3DCNN increased from the previous trial; however, it was insufficient to deem this training successful. This is evident from its loss function in Figure 35 and the confusion matrix in Figure 34, both detailed in Section B.4. The model is not effectively learning.

Conversely, the SlowFast Neural Network's performance deteriorated by 6.6% and showed no improvement when the batch size was increased. The loss function during training can be found in Figure 33, and the confusion matrix is presented in Figure 32, both available in Section B.3.

Training Dataset	**Val/Test Dataset**	**3DCNN**	SlowFast
MSASL	WLASL	14.58%	58.33%
WLASL	MSASL	3.60%	44.44%
WLASL increasing batch size and using Mixed Precision	MSASL	12.01%	37.84%

Table 6: Accuracy achieved by the 3DCNN and SlowFast models, trained using the WLASL dataset and evaluated on the MSASL dataset increasing the batch size to 10 and applying the mixed precision technique. Both datasets were previously processed using the ALL data processing type.

It is worth noting that increasing the batch size without employing Mixed Precision would likely result in performance improvement rather than degradation. However, due to GPU memory limitations, this test could not be conducted.

The code for this experiment can be found in the repository GitHub link.

3.3.4 Exploring training and testing dataset combinations

Despite being a commonly used model for video recognition with high accuracy expectations, the 3DCNN model did not perform as anticipated under

the same conditions as the SlowFast Neural Network. Consequently, a decision was made to switch to a different pre-trained model.

The selected pre-trained model, "i3d_r50" [35], was developed by Facebook and is accessible through PyTorch. This model, trained using the Kinetics dataset, demonstrated impressive performance in action recognition, boasting an accuracy of 74.3% [34].

This experiment also sought to identify the most suitable dataset for both training and testing. It explored whether the MSASL dataset was more appropriate for training and the WLASL dataset for testing, vice versa, or if a balanced combination of both for training and testing would yield the best results. The performance of the pre-trained model under these various dataset combinations is detailed in Table 7.

Given the larger size of the MSASL dataset, when used for training, it necessitated reducing the batch size to 5 and activating Mixed Precision, which in turn resulted in lower accuracy, as evidenced in Section 3.3.3. To maintain consistency across all trials in this experiment, the low batch size and Mixed Precision configurations were retained, even when the WLASL dataset was used for training.

Training Dataset	**Val/Test Dataset**	**3DCNN**	**SlowFast**
MSASL	WLASL	55.24%	36.36%
WLASL	MSASL	20.94%	32.53%
MIX	MIX	**62.84%**	**57.55%**

Table 7: Performance Comparison of 3DCNN and SlowFast Models Using Different Training and Testing Datasets

The performance of the pre-trained model was in fact influenced by the choice of training and testing datasets, as demonstrated in Table 7. It's noteworthy that the accuracy of the new pre-trained model improved compared to previous trials, rising from 14.58% (Table 6) to 20.94% (Table 7). Figures 38 and 39 in Section B.5.2 showcase the model's confusion matrix and loss function, respectively, during training with the WLASL dataset. These visualisations indicate that the model is indeed learning from the training set, despite the relatively low accuracy. However, this accuracy is insufficient to declare this model successful.

Conversely, when trained using the MSASL dataset, the 3DCNN's accuracy increased substantially to 55.25%, as evidenced by Figures 42 and 43 in Section B.5.4. The highest precision of 62.84% was achieved when both datasets were used for testing and training, as seen in Figures 46 and 47 in Section B.5.6. This result is remarkable, considering that the original model was achieving 74.3% accuracy during the training process and that the current model had not yet been optimised.

In the case of the SlowFast Neural Network, the reduction in batch size to 5 due to the necessity of conducting several tests in this trial resulted in a decrease in accuracy when training with the WLASL dataset. Accuracy dropped to 32. 53%, a decrease of approximately 12% compared to when the batch size was set to 10. These findings are illustrated in Figure 36 (confusion matrix) and Figure 37 (loss function), both available in Section B.5.1. This outcome suggests that increasing batch size, as observed in the previous experiment (Section 3.3.3), had a positive impact on the model by counteracting the decrease in accuracy caused by mixed precision.

However, when the MSASL dataset was used for training, the SlowFast accuracy increased to 36.36%. Figures 40 (confusion matrix) and Figure 41 (loss function), both located in Section B.5.3, demonstrate this improvement. The highest accuracy was achieved when both datasets were combined for training and testing, mirroring the 3DCNN results. This combination yielded an accuracy of 57.55%, as depicted in Figure 44 (confusion matrix) and Figure 45 (loss function), both accessible in Section B.5.5. This outcome aligns with expectations, as training in diverse situations, camera orientations, and with different interpreters likely contributes to improved model performance. Additionally, the training set becomes more similar to the test set when both datasets are used, potentially explaining the enhanced accuracy.

Notably, both models exhibited higher accuracy when the MSASL dataset was used for training rather than testing, which makes sense given its larger sample size. The code implemented for this experiment is available in the following repository: GitHub link.

3.3.5 Remove duplicated samples in the datasets

To further enhance the performance of both models, especially now that the 3DCNN is demonstrating the ability to learn and achieve a respectable ac-

curacy in line with its state-of-the-art capabilities, a detailed analysis of the confusion matrixs (Figures 44 and 46) was conducted. This analysis aimed to identify labels that were consistently misclassified.

Upon closer examination, it became evident that the SlowFast model struggled with two specific labels: "white" and "friends," neither of which it predicted correctly. To uncover what set these signs apart from the others, an investigation into the dataset revealed an intriguing issue within the MSASL dataset. Numerous samples in the MSASL dataset appeared to be duplicates, each assigned a unique video ID, despite being identical in content. This redundancy was causing overfitting, a phenomenon supported by the divergence between the validation and training loss functions (evidenced in Figures 45 and 47). In these figures, it is evident that the validation loss consistently exceeded the training loss by a significant margin.

Additionally, it was observed that the test set contained several videos that were identical to those in the training set. This explains the unexpectedly higher accuracy of both models, despite the absence of optimisation efforts.

To address this issue, a repetition of the previous experiment detailed in Section 3.3.4 was necessitated. This time, a manual analysis of each sample was conducted to identify and subsequently remove duplicated samples. The removal process adhered to the following conditions:

- Samples were removed if there was another video within the same dataset featuring the same signer signing the same label.
- Samples were retained if the same signer had multiple samples signing the same labels in distinct ways, a characteristic frequently observed in the WLASL dataset.
- Videos that contained several signs in one video, each lasting at least 30 seconds, were removed, as a sign typically lasts a maximum of 5 seconds.

Among the datasets, the MSASL dataset contained the highest number of duplicated videos, leading to the removal of a total of 186 videos. The labels most affected by this removal included:

- hungry: 7 samples were removed due to exact video duplicates.
- swing: 6 samples were removed due to exact video duplicates.
- want: 4 samples were removed due to exact video duplicates.

In the WLASL dataset, 27 videos were removed, with the most affected labels being:

- delicious: 3 samples were removed, including 2 exact duplicates and 1 from another video that was also identical.
- doctor: No samples were removed from this label. However, several videos were retained because the same interpreter interpreted the same sample in different ways, emphasising the importance of monitoring this label.
- drink: 2 samples were removed due to exact video duplicates.

Table 8 provides a detailed list of the identification numbers of all removed duplicated videos, serving as a reference for potential future repetitions of this experiment.

After the removal of all duplicated samples, the experiment detailed in Section 3.3.4 was re-executed, resulting in the outcomes displayed in Table 9. An analysis of these results reveals a noticeable decrease in accuracy across all model combinations. This outcome aligns with expectations, as the models were compelled to generalize their predictions rather than relying on memorized training set samples. Notably, even though both models experienced a decline in accuracy, the overall conclusions drawn in Section 3.3.4 remain valid. Specifically, when both datasets are employed for training, and the models' precision is divided into testing, superior performance is achieved compared to when either the MSASL or WLASL datasets are used for training. However, it's worth noting that when the MSASL dataset is used for training, the accuracy of both models drops significantly, which is expected due to the removal of a substantial number of samples from this dataset.

The corresponding accuracy values are presented in Table 9 for reference:

Analysing the most affected labels and comparing the confusion matrices before and after the removal of duplicated samples provides valuable insights

Dataset	Removed Samples
MSASL	['1001', '1002', '1004', '1027', '1031', '1042', '105', '1060', '1070', '1086', '1090', '1098', '1105', '1119', '1121', '1128', '1140', '1162', '1187', '1189', '1199', '1200', '1214', '1215', '1227', '1232', '1234', '1235', '1244', '1248', '1252', '1258', '1270', '1281', '1282', '1309', '1311', '1318', '1319', '1321', '1322', '1331', '1334', '1351', '1365', '1380', '1388', '1405', '1434', '1444', '1445', '1464', '1488', '1494', '1533', '1536', '1555', '1557', '1597', '1643', '1656', '1662', '1676', '1724', '1792', '1793', '1796', '1799', '1829', '1831', '1833', '1885', '1916', '1922', '1936', '1938', '1939', '1955', '1957', '1961', '1965', '2034', '2074', '2085', '2104', '2106', '2110', '2114', '2116', '2117', '2128', '2202', '2203', '2336', '2490', '255', '257', '296', '325', '327', '341', '342', '354', '37', '372', '380', '39', '390', '393', '402', '41', '44', '452', '471', '476', '480', '483', '507', '508', '51', '520', '521', '532', '534', '535', '536', '537', '538', '539', '542', '559', '560', '561', '566', '567', '569', '581', '582', '590', '595', '598', '614', '644.f136', '660', '662', '676', '677', '685', '689', '709', '729', '737', '738', '742', '753', '757', '807', '811', '818', '819', '822', '851', '862', '866', '883', '887', '900', '916', '917', '921', '922', '923', '924', '930', '941', '944', '946', '961', '967', '973', '974', '975', '976', '977', '978', '979']
WLASL	['07934', '07935', '09185', '14630', '15363', '15364', '15370', '17015', '17721', '17722', '21955', '22645', '32381', '33281', '36053', '36838', '36942', '41028', '50510', '51884', '51893', '56344', '57045', '62254', '62506', '63211', '70266']

Table 8: List of the identifications of all duplicated videos removed.

into the models' performance. This comparison is made between the scenarios where both datasets were used for training and testing, as they consistently yielded the best results and serve as the basis for further experiments. Refer to Figures 46 and 58 to compare the 3DCNN model's confusion matrices between the original experiment in Section 3.3.4 and the current one.

For the 3DCNN model:

- Hungry: In the original experiment, the model correctly predicted "hungry" 75% of the time (3 out of 4), but in the current experiment, it did not predict it correctly in any attempts.
- Swing: In the original experiment, "swing" was predicted correctly 100% of the time, but in the current experiment, the model made

Training dataset	**Val/Test Dataset**	**3DCNN**	SlowFast
MSASL	WLASL	29.20%	34.31%
WLASL	MSASL	45.58%	30.98%
MIX	MIX	**57.67%**	**47.62%**

Table 9: Performance Comparison of 3DCNN and SlowFast Models Using Different Training and Testing Datasets after Removing Duplicated Samples.

mistakes in 2 out of 5 attempts.

- Want: In the original experiment, one out of four samples was incorrectly predicted, while in the current experiment, the model made a mistake in 50% of the attempts.

Turning to the SlowFast Neural Network model, refer to Figures 44 and 56 for comparison:

For the SlowFast model:

- Hungry: Both experiments achieved the same 50% accuracy for this label, despite the removal of 7 samples.
- Swing: In the original experiment, "swing" was predicted correctly 80% of the time (4 out of 5), while in the current experiment, the model made mistakes in 2 out of 5 attempts. Consequently, the 3DCNN model was more accurate in the previous experiment, but in the current scenario, both models have the same error rate.
- Want: Surprisingly, the current experiment achieved higher accuracy when the duplicated samples were removed compared to the original experiment.

In particular, the removal of duplicated samples had a more pronounced impact on the 3DCNN model, especially for labels from which a significant number of samples were removed. However, both models experienced a decrease in precision from the previous experiment in Section 3.3.4 to the current one.

This outcome underscores the critical importance of maintaining a clean and representative dataset to ensure reliable model performance. For reference, the code for this experiment can be found in the following repository: GitHub link.

3.3.6 Results discussion

The primary objective of the third experiment was to enhance the performance of the 3DCNN model, as it exhibited limited learning capacity. The underlying idea behind this experiment was that the model's training dataset was insufficient, and supplementing it with additional data could improve its training process.

However, when the training dataset was expanded by incorporating the MSASL dataset, the 3DCNN model's performance continued to hover below the 20% accuracy threshold. Despite conducting numerous tests that involved varying the training and testing datasets, none of these attempts yielded satisfactory accuracy levels.

When it became evident that the initial intuition was incorrect, alternative methods were explored. Initially, the batch size was attempted to be increased, but computational constraints limited the extent to which this could be achieved. Ultimately, what allowed to achieve the goal of this experiment was the modification of the pre-trained 3DCNN model used. Additionally, after thorough data cleaning, the 3DCNN model achieved an accuracy rate of 57.67%.

Despite the initial misconception, this experiment successfully achieved its intended goal.

3.4 Experiment 4: Identify improvements in data processing

With both models now effectively learning and achieving high performance, the primary objective of the fourth experiment was to compare the data processing techniques employed. The aim was to assess whether these techniques were enhancing the model's accuracy, with a particular focus on the key parts of the interpreter's body where the sign language information is concentrated.

The rationale behind this experiment was rooted in the belief that the "FACE AND HANDS" and "BODY AND HANDS" data processing methods would have a more significant impact on the model's performance. This belief was based on the results obtained in the second experiment, as presen-

ted in Section 3.2.2 and Figure 3. While these results were initially observed with the SlowFast Neural Network, the intuition behind this experiment was that similar trends could be extrapolated to the 3DCNN model.

3.4.1 Data selection and processing

The dataset is categorised into three distinct groups based on the location of the sign movement within the interpreter's body. These categories include scenarios where the movement is solely concentrated in the hands, or it encompasses both facial expressions and movements of the arms and shoulders. Table 4 provides a breakdown of which category each label used for training falls into.

To comprehensively assess the impact of the four distinct data processing techniques described in Section 3.2.2, the label categories were trained using all available data processing methods. For instance, if the "FACE AND HANDS" processing type proves effective, it would be expected to yield higher accuracy for labels associated with facial gestures compared to other data processing methods.

To understand the computational cost of this experiment, Table 10 details the time required to process all samples with the "FACE AND HANDS" method. The other methods took a similar amount of time, except for the "HANDS," which was completed in half the time.

Dataset	**Time Cost (in hours)**
MSASL	09:49:15
WLASL	06:34:45

Table 10: Time required to process the 50 samples with the FACE AND HANDS processing method for each dataset

3.4.2 Results

Table 11 presents the outcomes of the current experiment, with the highest accuracy achieved among the data processing techniques for each label cat-

egory highlighted in bold.

Labels	**Data Processing**	**3DCNN**	**SlowFast**
All	All	**68.25%**	48.15%
All	Face and Hands	66.88%	**50.00%**
All	Body and Hands	65.08%	49.21%
All	Hands	51.49%	37.62%
Hands gestures	All	**74.64%**	**63.04%**
Hands gestures	Face and Hands	70.27%	27.03%
Hands gestures	Body and Hands	71.01%	52.17%
Hands gestures	Hands	50.65%	46.75%
Face gestures	All	**70.37%**	59.26%
Face gestures	Face and Hands	52.17%	34.78%
Face gestures	Body and Hands	66.67%	**62.96%**
Face gestures	Hand	50.00%	41.67%
Body gestures	All	**87.50%**	83.33%
Body gestures	Face and Hands	75.00%	70.00%
Body gestures	Body and Hands	83.33%	**87.50%**
Body gestures	Hands	75.00%	41.67%

Table 11: Summary of accuracy results for the SlowFast Neural Network and 3DCNN trained with four different data processing techniques across three label categories and all labels combined.

Analysing it by parts, starting with when all labels are considered in the training process. For the 3DCNN model in this context, the highest accuracy is attained when using the "ALL" data processing type (refer to Figures 62 and 63 in Section C.1.2 for the confusion matrix and loss function, respectively). Notably, the "FACE AND HANDS" and "BODY AND HANDS" processing types yield very similar accuracy rates, differing by only around 3%. The "HANDS" data processing, on the other hand, exhibits a significantly lower accuracy, which is expected, as it contains less information and is trained across all labels, including those requiring facial expressions and body movements.

For the SlowFast Neural Network, a similar pattern emerges, with the highest accuracy attributed to the "FACE AND HANDS" data processing technique (refer to Figures 60 and 61 in Section C.1.1 for the confusion matrix and loss function, respectively). The lowest accuracy is associated with the "HANDS" processing, displaying a 10.53% difference from the third-highest

accuracy level.

When considering only the "hands gestures" labels for training on both models, the highest accuracy is consistently achieved using the "ALL" data processing method. For detailed visualisations of the 3DCNN model's confusion matrix and loss function, refer to Figures 66 and 67 in Section C.2.2, and for the SlowFast Neural Network model's equivalents, see Figures 64 and 65 in Section C.2.1. The "HANDS" processing method ranks as the third-best choice for the SlowFast Neural Network and fourth for the 3DCNN. This suggests that even when the sign movement is primarily located in the hands, it is insufficient to capture all the sign details. This outcome aligns with the observation that even in hand-focused signs, there may be additional movement in other parts of the body, such as the arms. A surprising finding is the closely matched accuracy of the "FACE AND HANDS" and "BODY AND HANDS" processing techniques when employed with the 3DCNN. This is unexpected, as it might have been assumed that facial features would not provide relevant information in signs where facial expressions are not required. Notably, the SlowFast Neural Network, has the lowest overall accuracy in the table when using the "HANDS" data processing over the hand gestures.

Moving on to "face gestures," a curious trend emerges. While high accuracies are achieved when using all labels, hand gestures, and body gestures, the accuracy drops significantly when training specifically for labels that require facial expressions. This decline may be attributed to the subtle and brief nature of facial expressions in sign language. Interestingly, the SlowFast Neural Network appears to effectively capture these subtle facial expressions, as it performs best when all labels are used. This outcome is expected, considering that the SlowFast Neural Network operates with two channels, one of which analyses low-frequency changes between frames. Intuitively, one might have assumed that this processing would align well, especially for facial gestures. For detailed insights and visualisations of the highest accuracy results, please consult Figures 70 and 71 in Section C.3.2. Similarly, the SlowFast Neural Network model's equivalent visualisations can be found in Figures 68 and 69 in Section C.3.1.

Finally, turning the attention to "body gestures" labels, the results follow a predictable pattern. The highest accuracy is achieved when employing the "BODY AND HANDS" data processing method with the SlowFast Neural Network. For a detailed breakdown of the SlowFast Neural Network's performance, please refer to Figures 72 and 73 in Section C.4.1, where the model's confusion matrix and loss function are available. On the other hand,

with the 3DCNN model, the highest accuracy is attained using the "ALL" data processing method. For in-depth insights into the 3DCNN model's performance, including the confusion matrix and loss function, please consult Figures 74 and 75 in Section C.4.2. In summary, for "body gestures," the SlowFast Neural Network achieves its highest accuracy with the "BODY AND HANDS" data processing method, followed by the "ALL" data processing method. Conversely, the 3DCNN model performs best with the "ALL" data processing method, followed by the "BODY AND HANDS" processing type.

In conclusion, it appears that the most consistently effective data processing method is "ALL." For the 3DCNN, "ALL" consistently yields the highest accuracy, followed by "BODY AND HANDS," with "FACE AND HANDS" as the third-best choice. In the case of the SlowFast Neural Network, the highest accuracy is observed with both "ALL" and "BODY AND GESTURES." Throughout the experiment, the "HANDS" processing type consistently under-performs, even when applied exclusively to "hands gesture" labels. Face gestures do not seem to benefit significantly from the "FACE AND HANDS" processing type, except when considering all labels, as observed in the SlowFast Neural Network.

The GitHub link hosts the code for the fourth experiment.

3.4.3 Results discussion

The objective of this experiment was to enable a comparison of the different data processing techniques outlined in Section 3. By training both models using the four types of processing across four different data selections, it became possible to discern which technique provided the models with the highest information content about the sign, leading to the highest accuracy.

The results consistently ranked the "ALL" processing technique as the most impactful and reliable, consistently placing it at the top for both models. The "BODY AND HANDS" and "FACE AND HANDS" processing method exhibited a similar degree of relevance, although with somewhat less consistency. Notably, the "HANDS" data processing technique consistently performed the poorest among all the tests.

Initially, there was an intuition that the "BODY AND HANDS" and

"FACE AND HANDS" techniques would yield the highest accuracy, influenced by the results obtained in Section 3.2.2. While both methods are indeed effective, the "ALL" processing technique outperformed them in this experiment. It's important to note that this outcome may differ due to variations in labels, label quantity, and, most importantly, the 3DCNN model being used.

3.5 Experiment 5: Models optimisation

In the previous experiments, the 3DCNN achieved the highest accuracy at 68.25%, while the SlowFast Neural Network reached a maximum accuracy of 50%, both trained with 50 different labels.

The objective of this experiment is to enhance the accuracies of both models. The intuition behind this effort is that by fine-tuning the model parameters, their performances can be improved. This fine-tuning would adapt the models to the specific characteristics of the input data for the current problem, rather than relying on general parameter settings.

3.5.1 Optimisation specifications

Due to computational constraints related to timing, memory, and the extensive nature of the experiments, a careful selection of feasible parameter values was necessary. Table 12 presents the parameters chosen and their respective tested values. Both models were tested with all possible combinations of these values.

For this experimentation, the dataset comprising 50 labels and processed using the ALL processing method was chosen. This selection was based on its reliability, as demonstrated in the previous experiment (Section 3.4), where it was determined to be the most robust option. Given the considerable time required for the experiment to execute, spanning several days, it was not feasible to test it across multiple data processing techniques.

Table 13 presents the parameter values that were applied to both the 3DCNN and SlowFast Neural Network models in the experiments conducted prior to the current one. This table will serve as a reference point for com-

Parameters	Possible Values
Learning rate	[0.001, 0.01, 0.1]
Momentum	[0.4, 0.6, 0.9]
Weight decay	[0.0001, 0.001, 0.01]
Loss function	["Cross Entropy", "Log-Likelihood"]
Optimiser	["Adam", "SGD"]

Table 12: The values that were tested for each of the parameters to optimise.

paring the optimal values obtained in this experiment with those utilised up to this point.

Parameters	3DCNN	SlowFast Neural Network
Learning rate	0.001	0.1
Momentum	0.4	0.3
Weight decay	0.01	0.01
Loss function	Cross Entropy	Cross Entropy
Optimiser	SGD	SGD
Accuracy	68.25%	50%

Table 13: Parameter values for the models used in all the experiments preceding the current one.

3.5.2 Results

Figure 6 provides a visual representation of how the accuracy of the models fluctuates when exposed to different learning rate values during the experiment. In this specific study, three distinct learning rates were considered: 0.001, 0.01, and 0.1. Each of these values signifies a unique rate at which the model adjusts its internal parameters during the training process.

The graph's x-axis corresponds to these learning rate values, while the y-axis represents the model's accuracy. The primary objective of this experiment was to pinpoint the learning rate value that yields the highest accuracy for each model.

To facilitate interpretation, the purple cross on the graph highlights the point at which the 3DCNN achieved its peak accuracy, while the red cross indicates the same for the SlowFast Neural Network. Both models attained

their highest accuracy when the learning rate was set to 0.01.

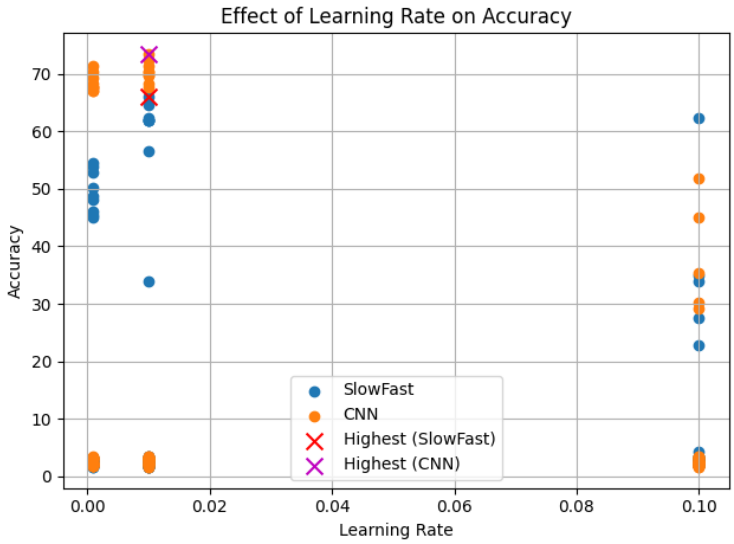

Figure 6: Accuracy variance diagram for 3DCNN and SlowFast models across different learning rate parameters.

Figure 7 provides insights into how the model's accuracy fluctuates when subjected to variations in the momentum parameter. For this experiment, three momentum values were tested: 0.4, 0.6, and 0.9. In the case of the 3DCNN model, the highest accuracy was achieved when the momentum was set to 0.6. Conversely, for the SlowFast Neural Network, the peak accuracy was obtained with a momentum value of 0.9. This visualisation helps in identifying the optimal momentum parameter settings for each model.

The graph in Figure 8 shows how the accuracy of the model changes when the weight decay parameter is varied. Three momentum values were tested: 0.0001, 0.001 and 0.01. The 3DCNN model achieved the highest accuracy when the momentum was set to 0.0001, while the SlowFast Neural Network had the highest precision with a momentum value of 0.001. This graph helps to determine the best momentum parameter settings for each model.

Figure 9 displays the impact of the loss function on the accuracy of the model. Two distinct loss functions were evaluated: Cross Entropy and Log-Likelihood. Surprisingly, the Log-Likelihood loss function consistently resulted in significantly lower accuracy, with a maximum of 3.5% achieved in both

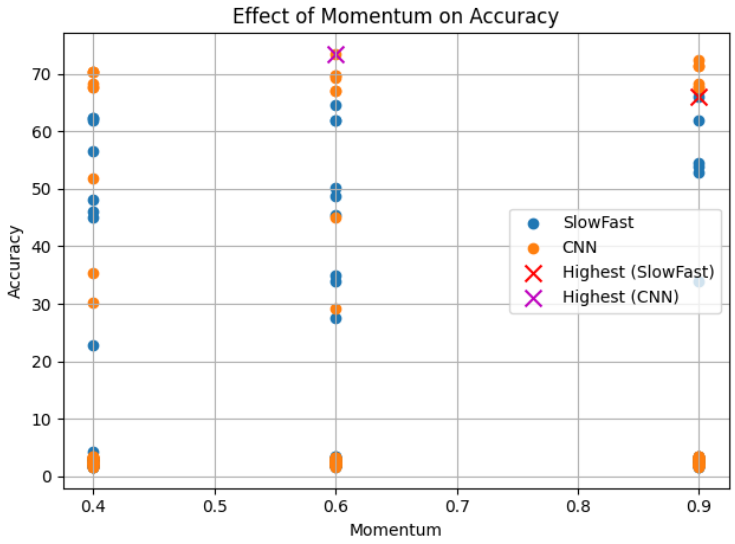

Figure 7: Accuracy variance diagram for 3DCNN and SlowFast models across different momentum parameters.

models. This clear contrast in performance indicates that the Cross Entropy loss function is the more appropriate choice for this task.

Finally, Figure 10 illustrates the influence of the optimiser on the accuracy of the model. Two different optimisers were assessed: SGD and Adam. Interestingly, Adam, optimiser, consistently yielded notably lower accuracy, with a maximum of 3.5% achieved in both models. This stark difference in performance underscores the suitability of the SGD optimiser for this particular task.

After applying the optimised parameter values, the results shown in Table 14 were achieved. It's noticeable that there is a slight drop in accuracy when transitioning from the validation set to the test set; however, both models maintain an accuracy of over 60%. In the case of the SlowFast Neural Network, the accuracy on the test set improved from 50% to 62.43%, representing a significant 12.43% enhancement resulting from this optimisation experiment.

Conversely, the 3DCNN's accuracy saw a slight decrease, declining from 68.25% to 67.72%, which, although marginal, is still noteworthy. It's important to note that due to the computational constraints within this project, only a limited range of parameter values could be explored. This implies

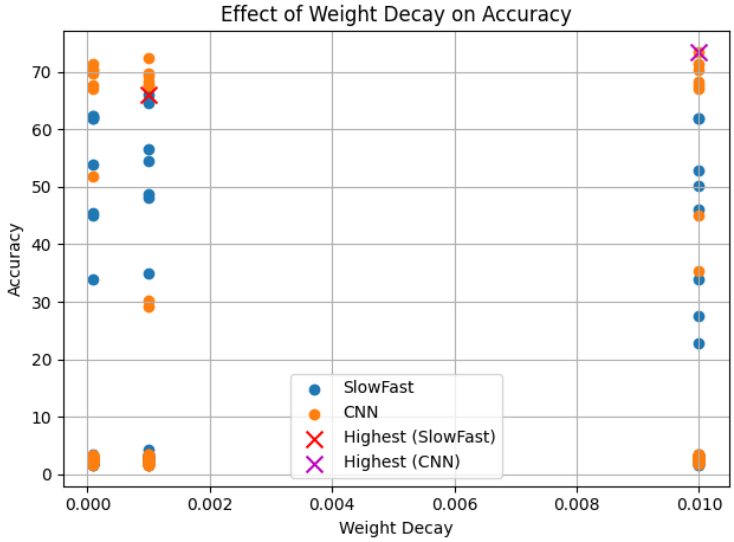

Figure 8: Accuracy variance diagram for 3DCNN and SlowFast models across different weight decay parameters.

that there may be other combinations of parameters that could yield even higher accuracy. However, considering the project's constraints, the achieved performances of both models are remarkable.

SET	**3DCNN**	**SlowFast Neural Network**
Validation	73.54%	66.13%
Test	67.72%	62.43%

Table 14: Accuracy obtained training both models using the optimised parameters on the validation and test sets.

Table 15 presents a concise summary of the parameter values that resulted in the highest accuracy for each model after the optimisation process. In particular, values that have been modified from previous executions are highlighted in bold. To better understand the changes made during optimisation, refer to Table 13 for a comparison with the previous parameter values.

Once the optimised parameters were determined, the models underwent retraining using these optimised configurations along with the three data processing methods that exhibited the highest performance in the fourth experiment, as discussed in Section 3.4. Table 16 displays the resulting accuracies.

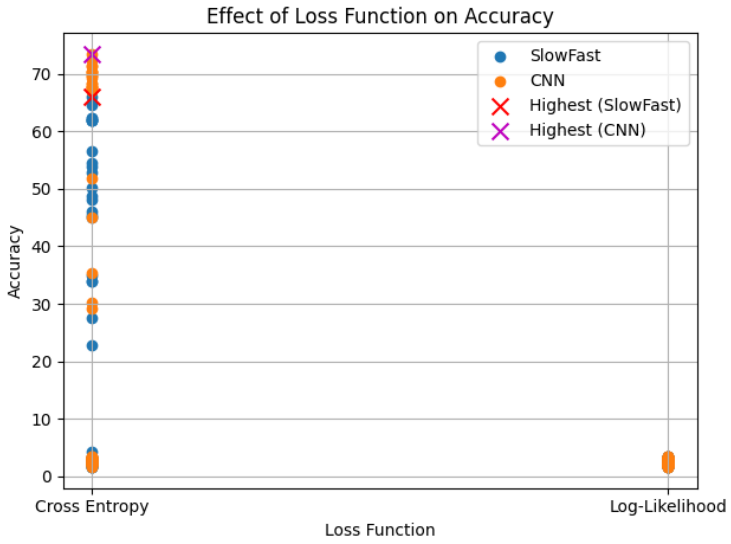

Figure 9: Accuracy variance diagram for 3DCNN and SlowFast models across different loss functions.

Parameters	**3DCNN**	**SlowFast Neural Network**
Learning rate	**0.01**	**0.01**
Momentum	**0.06**	**0.09**
Weight decay	0.01	**0.001**
Loss function	Cross Entropy	Cross Entropy
Optimiser	SGD	SGD
Accuracy	67.72%	62.43%

Table 15: Optimised values for the parameters of the 3DCNN and SlowFast Neural Network models.

The SlowFast Neural Network demonstrated improved performance across all processing types, whereas the 3DCNN exhibited a decline in accuracy for all of them. These results suggest that the previous 3DCNN model's configuration was better suited for maximising model performance. Consequently, the parameters from Table 13 will continue to be utilised for the 3DCNN.

The code for the fourth experiment is located at the GitHub link repository.

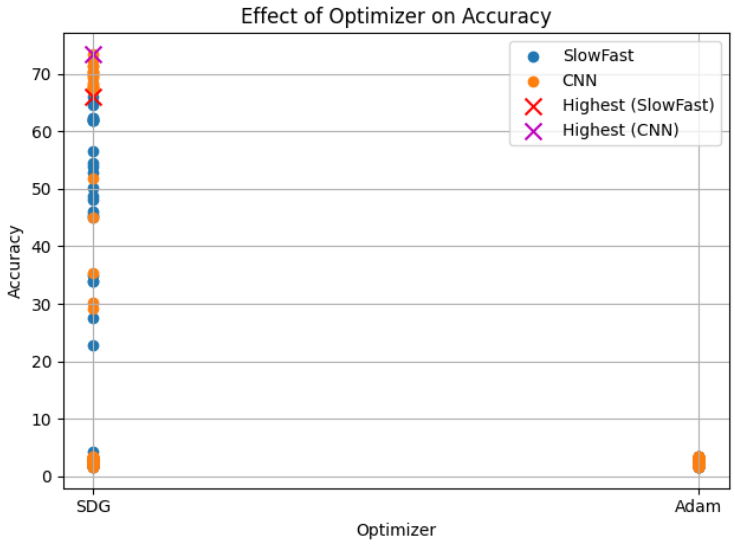

Figure 10: Accuracy variance diagram for 3DCNN and SlowFast models across different optimisers.

PROCESSING TYPE	**3DCNN**	**SlowFast**
ALL	67.72%	62.43%
BODY AND HAND	55.03%	55.56%
FACE AND HANDS	41.56%	66.23%

Table 16: Accuracies achieved after optimising the parameters for the 3DCNN and SlowFast Neural Network models.

3.5.3 Results discussion

The primary goal of the current experiment was to enhance the performance of both models by fine-tuning their parameters to achieve optimal accuracy on the validation dataset.

In the case of the SlowFast Neural Network model, this objective was successfully accomplished. A comparison between the results presented in Table 11 and the outcomes of the current experiment displayed in Table 16 clearly illustrates how the model's performance has improved across all processing methods.

Conversely, the 3DCNN model experienced a decline in performance when using the optimised parameters. Due to computational limitations, it was

not feasible to exhaustively test all possible parameter combinations. Consequently, some of the optimised parameters resulted in lower accuracy compared to the initial model configuration.

3.6 Experiment 6: Implement ensemble model

The final experiment is designed to create an ensemble model with the objective of enhancing the current performance of both the 3DCNN and the SlowFast Neural Network. This ensemble model combines different models, each trained on various data processing methods of both the 3DCNN and the SlowFast Neural Network.

The motivation for this experiment stems from the observation that the individual models achieve accuracies exceeding 60%. By constructing an ensemble model, the aim is to achieve an increase in accuracy by capitalising on the distinct strengths of each model. Examining the confusion matrices in the appendix (Section C), it is evident that these models excel in different label predictions. By aggregating the outputs of these models using a weighted function, the goal is to utilise their individual strengths to enhance overall accuracy.

3.6.1 Ensemble model architecture

The code allows the ensemble model to incorporate either three 3DCNN models or three SlowFast Neural Network models. The architecture of the ensemble model is illustrated in Figure 11. There are three distinct inputs, each containing the 50 different labels, differentiated solely by the applied data processing technique. The first model takes as input the samples processed using the "ALL" method, which had been previously trained using the same processing type on the training set. The second model was trained with samples processed using the "FACE AND HANDS" technique, while the third model utilized the "BODY AND HANDS" processing method. Although the input samples were the same (only processed differently), the models generated predictions independently. These predictions were then combined and fed as input to a Multilayer Perceptron (MLP) model, which calculated weights to determine the reliability of each model's output. The

final output of the ensemble model was produced by the MLP.

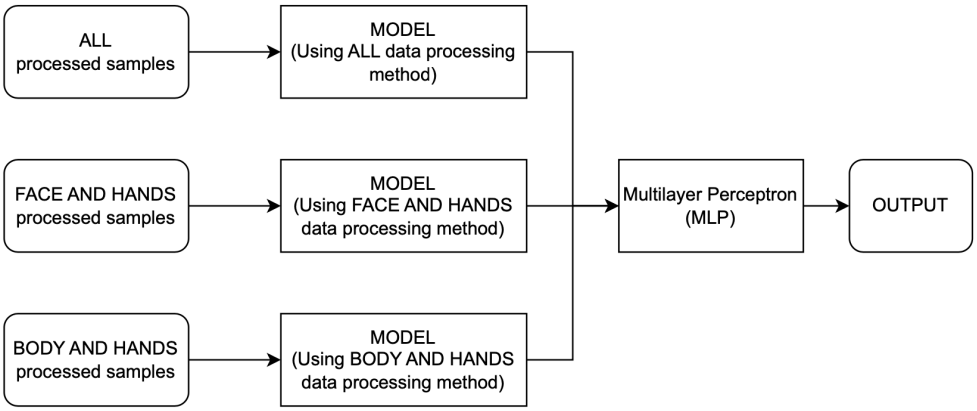

Figure 11: Illustration depicting the architecture of the proposed ensemble model. The chosen models for inclusion in the ensemble can be either the SlowFast Neural Network or the 3DCNN models, and they share a common high-level architecture.

3.6.2 Results

The model defined in Section 3.6.1 was trained using the SlowFast Neural Network models from Experiment 5 in Section 3.5, while the 3DCNN models were obtained from Experiment 4 in Section 3.4.

Numerous tests were conducted, primarily due to the significant computational cost required for these models. Initially, the output feature maps from the trained models were passed to the MLP to provide more information for predictions, including not just the top-1 result. However, this approach proved inadequate as both executions were halted due to a CUDA Out Of Memory Error.

In response to this limitation, the next attempt aimed to mitigate the memory issue by forwarding only the predicted label, thereby reducing the input size. Despite this adjustment, the problem persisted as the memory constraints remained unresolved, preventing successful execution.

The third strategy involved storing the three trained models in CPU memory and then sending their outputs to the GPU, allowing for the ef-

ficient training of the MLP model with CUDA acceleration. This method was only feasible for the 3DCNN model, as the SlowFast Neural Network encountered an Out Of Memory error. While this approach did yield results, the final output accuracy reached only 3.45%, as the small batch size required to accommodate memory limitations hindered the model's capacity to learn effectively.

In a fourth attempt to address the issues, the ensemble model was configured to receive the feature array as input once again, providing more data to the model while keeping the batch size small to avoid memory constraints. Unfortunately, even with the three trained models residing in GPU memory, the Out Of Memory Error remained unresolved.

Finally, in a fifth iteration, an alternative approach was explored. The trained models' predictions were stored in an external file, eliminating the need to store the models themselves in GPU memory during MLP training. However, this strategy also faced difficulties as the execution was terminated due to a memory leak detected in the GPU memory.

3.6.3 Results discussion

The experiment outlined in this section held significant potential for improving the models' performance. However, due to the severe computational limitations encountered, it could not be executed as intended. These limitations were persistent throughout the entire research project, but they became particularly pronounced in this experiment. The ensemble method involved not only using four different models but also storing the same sample size multiplied by three, which significantly exacerbated the memory and computational demands.

Although the results of this section are unavailable, it is highly advisable to explore and attempt this approach if computational resources are not a limiting factor. This experiment represents an intriguing avenue for further research and potential improvement in sign language recognition models.

Despite the absence of results, all the code implementations from the different experiments and iterations can be found in the following repository: GitHub link. This repository provides valuable insights and a foundation for future researchers to build upon, potentially unlocking new approaches for

sign language recognition.

3.7 Results discussions across all experiments

The primary objective of this section was to conduct a series of experiments aimed at developing a Sign Language Recognition (SLR) model capable of accurately predicting English words from video recordings of American Sign Language (ASL) interpretations.

While the resulting models did not surpass the state-of-the-art in SLR, these experiments have yielded valuable insights into the factors influencing model performance and the interpretation of sign videos.

One key observation was the temporal relevance within the sign video. It became evident that the start and middle portions of the sign held more significance for accurate prediction, while the final segments was less relevant. Additionally, increasing the batch size had a positive impact on model accuracy. However, due to computational constraints, the batch size was limited, suggesting that models trained in a more resource-rich environment could achieve better results.

The impact of different data processing techniques on model performance was also explored. These techniques included determining which parts of the videos carried the most valuable sign information, such as the face, hands, entire frame, or combinations of these. Notably, relying solely on hand information proved insufficient, as sign language interpretation often involves multiple body parts.

The use of multiple datasets further enriched the model's generalisation capabilities, despite the challenges posed by data processing and cleaning. The diversity of data sources provided a broader perspective on sign types and interpretations.

The importance of clean datasets was underscored, as similarities between training and test data can lead to overfitting. Various dataset configurations were experimented with, ultimately finding that a combination of datasets for both training and testing improved model performance. Proper dataset evaluation was emphasised to ensure data accuracy.

To optimise model performance, parameter tuning was conducted, adjusting the models to suit the specific problem and input data. This optimisation notably increased the accuracy of the SlowFast Neural Network by approximately 12%.

Finally, ensemble modeling, a promising approach that combines predictions from multiple models to enhance overall performance, was explored. While computational limitations prevented obtaining conclusive results, the concept remains an intriguing avenue for future SLR research.

All the code and documentation for each experiment are publicly available in the GitHub repository, allowing for easy replication and further exploration. Despite the challenges faced, these experiments have contributed valuable knowledge to the field of Sign Language Recognition.

4 Results

In Section 3, various experiments were conducted, each with a distinct objective. However, all these experiments shared a common goal: the development of a high-performing Sign Language Recognition model that could be bench marked against the state-of-the-art.

Table 17 provides an overview of the best accuracies achieved by the two implemented models. The 3DCNN achieved its highest accuracy peak during the fourth experiment in Section 3.4, with an accuracy rate of 68.25%. This model was trained using 50 labels (please refer to Table 4 for the specific labels it was trained on) and employed the ALL data processing technique, as explained in Section 3.2.2. In this technique, both face and hand detection were performed on each frame of the input data, resulting in an image constructed from four regions: the left and right hands at the top, and the face and a blurred original image at the bottom. For a visual representation, please consult Figure 2.

The SlowFast Neural Network achieved its peak accuracy of 66.23% during the fifth experiment in Section 3.5. Similarly, it was trained using the same 50 labels. However, in this case, the optimal data processing method was FACE AND HANDS, where the original, blurred image was excluded from the input images. Instead, the bottom region of the constructed image exclusively featured the facial expressions of the sign interpreter. For a visual representation of the input data used, please consult Figure 3.

PROCESSING TYPE	**3DCNN**	**SlowFast**
ALL	**68.25%**	62.43%
BODY AND HAND	65.08%	55.56%
FACE AND HANDS	66.88%	**66.23%**

Table 17: This table summarises the highest accuracy achieved for both the 3DCNN and SlowFast Neural Network, considering the top three performing data processing methods.

When comparing the performance of both models using Table 17, it becomes evident that the 3DCNN consistently outperformed the SlowFast Neural Network across all processing types. Although the FACE AND HANDS method shows only a marginal difference between both models, it represents

the highest accuracy achieved by the SlowFast, while the 3DCNN exhibits an even stronger performance when utilising the ALL technique.

These results underscore the superiority of the 3DCNN over the SlowFast Neural Network in the current experimental setup. However, it's essential to consider that these experiments were conducted under computational limitations. It's plausible that the SlowFast could surpass the 3DCNN's performance with access to a more substantial batch size.

Additionally, it's noteworthy that the SlowFast Neural Network achieved an accuracy of 77% when trained on the Kinetics dataset for action recognition purposes, as documented in [34]. Furthermore, in the context of Sign Language Recognition (SLR) using the WLASL dataset [23], it demonstrated an accuracy of 79.34%. On the other hand, the 3DCNN model, when applied to action recognition with the Kinetics dataset, attained an accuracy of 74.3% [34]. It's worth mentioning that, in a state-of-the-art setup, the 3DCNN was combined with an MLP across 40 classes archiving an accuracy of 84.38%, as reported in [22].

Considering the constraints imposed by computational resources during the experiments, the achieved accuracies are remarkably close to the state-of-the-art. It's important to emphasise that with access to the requisite computational capabilities, the proposed models have the potential to deliver groundbreaking results. This optimism is rooted in the extensive experimentation and data processing analyses that have been conducted to optimise the models' performance.

5 Project Management

The project has been managed and monitored through a dedicated GitHub repository, which is currently accessible for public viewing. For implementation purposes, a personal computer and the university's provided servers have been used to leverage additional computational capacity.

Supervision of the project was conducted by Dr. Ahir Bhalerao, with regular meetings scheduled every two weeks. These meetings serve as opportunities to review progress, discuss new developments, analyse results, and plan the next steps of the project. These regular meetings with the supervisor ensure effective guidance and facilitate a smooth progression of the research work.

5.1 Example of Execution

Below are examples of executing commands for various tasks within the Sign Language Recognition project, including downloading and processing datasets, training and evaluating the 3DCNN SlowFast Neural Network and ensemble models.

5.1.1 Download and Process Datasets

To prepare the datasets for training and evaluation, follow these steps:

Example of executing commands to download and process video samples from the WLASL dataset:

```
$ python3.9 processing/video_downloader.py
$ python3.9 processing/process_videos.py --type
    face_and_hands
$ python3.9 processing/split_datasets.py
```

5.1.2 Train and Evaluate a 3DCNN

To train and evaluate the 3DCNN model, use the following commands:

To train and evaluate the model:

```
$ python3.9 train_model.py --model 3dcnn --eval True
  --data "/Document/processed_data" --checkpoint "/
  Documents/check_points"
```

To train without evaluation:

```
$ python3.9 train_model.py --model 3dcnn --eval False
  --data "/Document/processed_data" --checkpoint "/
  Documents/check_points"
```

To evaluate the trained model:

```
$ python3.9 evaluate_model.py --model 3dcnn --data "/
  Document/processed_data" --checkpoint "/Documents/
  check_points" --file "/Documents/check_points/3
  dcnn_model.pth"
```

5.1.3 Train and Evaluate a SlowFast Neural Network Model

For training and evaluating the SlowFast Neural Network model, use the following command:

To train and evaluate the SlowFast model:

```
$ python3.9 train_model.py --model slowfast --eval
  True --data "/Document/processed_data" --
  checkpoint "/Documents/check_points"
```

5.1.4 Train and Evaluate the Ensemble Model

To train and evaluate the ensemble model, follow these steps:

Example of executing commands to train and evaluate an ensemble model:

```
$ python3.9 train_model_ensemble.py --eval True
```

These examples demonstrate the workflow for various tasks in the Sign Language Recognition project, from data preprocessing to model training and evaluation.

6 Appraisal and reflection

Throughout the course of the experiments (as detailed in Section 3), several notable challenges were encountered.

Foremost among these challenges were the computational limitations. Hardware constraints necessitated resourceful solutions. While the university provided resources for conducting experiments, each student had access to limited storage space. Given that the dataset primarily comprised videos, which consume more storage compared to images, this posed a significant hurdle. Following multiple discussions with the university's IT team, an increase in storage capacity was secured. This expansion enabled the training of models on remote servers.

However, time constraints also came into play. With a project submission deadlines and a need for multiple experiments to yield promising results, the CPU memory upgrade alone proved insufficient. The models required GPU storage for faster training. Unfortunately, GPU resources were severely constrained and could not be expanded. This predicament restricted the number of experiments that could be executed. Notably, experiments aren't always executed once; they may need to be rerun due to errors or to explore different parameter configurations. Memory limitations significantly impacted the project's development.

Another challenge emerged when working with pre-trained models. Some research papers lacked comprehensive details about model architecture and expected input data formats. This lack of clarity made retraining the models and understanding the required input data format a complex task. It necessitated multiple attempts and extensive investigation to achieve successful training.

In addition, it is often challenging to locate pre-trained models as they are not always readily available. Many research papers do not provide specific details or direct links to the pre-trained models they utilised. Luckily, when a paper employs a pre-trained model, they typically cite the source from which it was obtained. This citation serves as a helpful reference, facilitating the search for suitable pre-trained models that are accessible for download.

Despite these challenges, proactive measures were taken to overcome them

and ensure the smooth progression of the experiments.

7 Ethics

The WLASL [25] and MSASL [26] datasets utilised in this study involve the secondary analysis of publicly available data. The creators of these datasets explicitly specify that all data is intended solely for academic and computational purposes. As this report employs the dataset for academic research and adheres to proper citation practices, there is no breach of ethical consent.

8 Future work

This project successfully conducted several experiments, achieving performance close to the state-of-the-art with two distinct models: the 3DCNN and the SlowFast Neural Network. However, there remain potential avenues for further experimentation to enhance their performance and expedite progress toward creating a more inclusive society that supports individuals with speech and hearing impairments.

One primary challenge throughout this project was computational limitations. To address this, the proposed models could be retrained with increased batch sizes and a higher number of epochs. This adjustment would provide the models with better training conditions, potentially leading to improved performance.

With access to more robust computational resources, Experiment 3.6 could be executed to test its initial hypothesis. This experiment involves implementing an ensemble model that combines the best three models of each model type, distinguished by the data processing techniques used for input. The predictions from these models are then fed into a Multi-Layer Perceptron (MLP) to obtain a weighted combination of their outputs. Another intriguing approach would be to create a similar ensemble model but utilising both the SlowFast and the 3DCNN, with both model outputs entering the MLP. This combination of these promising models could provide valuable insights and potentially surpass their current accuracy levels.

Additionally, due to memory limitations, the dataset size had to be constrained. It would be beneficial to apply data augmentation techniques to expand the dataset, providing the models with a broader understanding of potential signs and improving their overall performance.

In conclusion, this project has made substantial progress in Sign Language Recognition, approaching state-of-the-art performance with the 3DCNN and SlowFast Neural Network models. Despite computational limitations, experiments have laid a strong foundation for future advancements. By addressing these constraints through larger batch sizes, extended training epochs, ensemble models, and data augmentation, the field can continue its journey towards inclusivity for individuals with speech and hearing impairments.

9 Conclusion

In conclusion, this project has made significant strides in the field of Sign Language Recognition (SLR) by conducting rigorous experiments and implementing advanced models. The primary focus was on two models: the 3DCNN and the SlowFast Neural Network, with the overarching goal of advancing the accuracy of Sign Language Recognition.

The experimentation encompassed several critical phases, with a primary emphasis on data processing techniques and model optimisation. Various data processing methods were rigorously evaluated, taking into account the critical information's location during sign interpretation. For each frame of every video sample, face and hand detection techniques were meticulously executed to reconstruct the input data, placing greater importance on these essential features.

The models were meticulously trained using a set of 50 manually selected labels. These labels were chosen not only to represent the most significant words used in daily life but also to mirror the different aspects of sign language. Approximately 68% of the signs primarily involved hand movements, 16% relied on facial expressions, and the remaining portion pertained to signs necessitating whole-body movements, such as arm and shoulder gestures.

Throughout these experiments, noteworthy achievements were attained, with the 3DCNN model achieving an accuracy of 68.25%, and the SlowFast Neural Network reaching 66.23%. It's important to highlight that these accomplishments were realised under challenging constraints, including limited computational resources and memory restrictions.

Looking ahead, the project lays a strong foundation for future work. Further research could involve refining the models by retraining with larger batch sizes and more epochs, exploring ensemble models that combine the strengths of the 3DCNN and SlowFast networks, and applying data augmentation techniques. These future endeavours aim to propel Sign Language Recognition technology even further, fostering inclusivity by providing effective communication tools for individuals with speech and hearing impairments.

A Second experiment

A.1 SlowFast neural network with ALL data processing

Figure 12 portrays the confusion matrix for the SlowFast neural network of the second experiment using the "ALL" data processing technique. Additionally, Figure 13 depicts the model's loss function during training. This particular model attained an accuracy level of 58.33%.

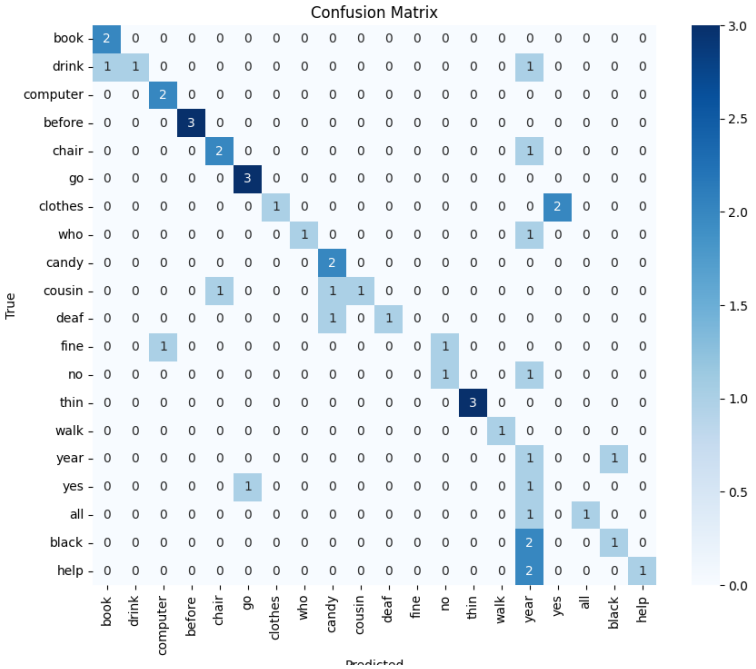

Figure 12: The confusion matrix for the second experiment's SlowFast neural network, employing the "ALL" data processing technique.

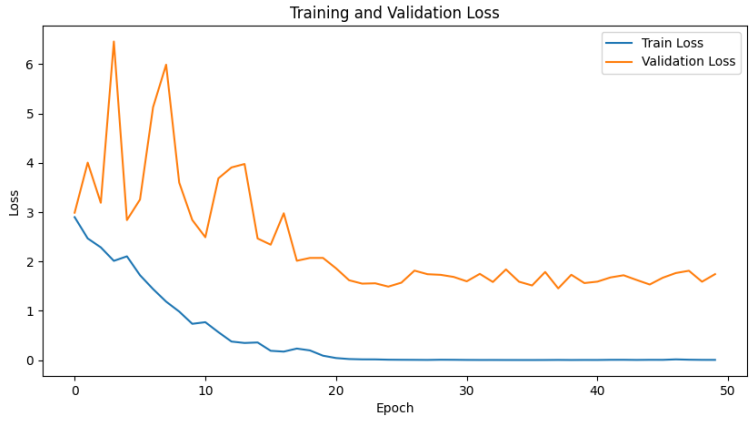

Figure 13: The loss function graph obtained from training the SlowFast neural network in the second experiment, employing the "ALL" data processing technique.

A.2 3DCNN with ALL data processing

Figure 14 shows the confusion matrix for the 3DCNN of the second experiment using the "ALL" data processing technique. Additionally, Figure 15 depicts the model's loss function during training. This particular model attained an accuracy level of 14.58%.

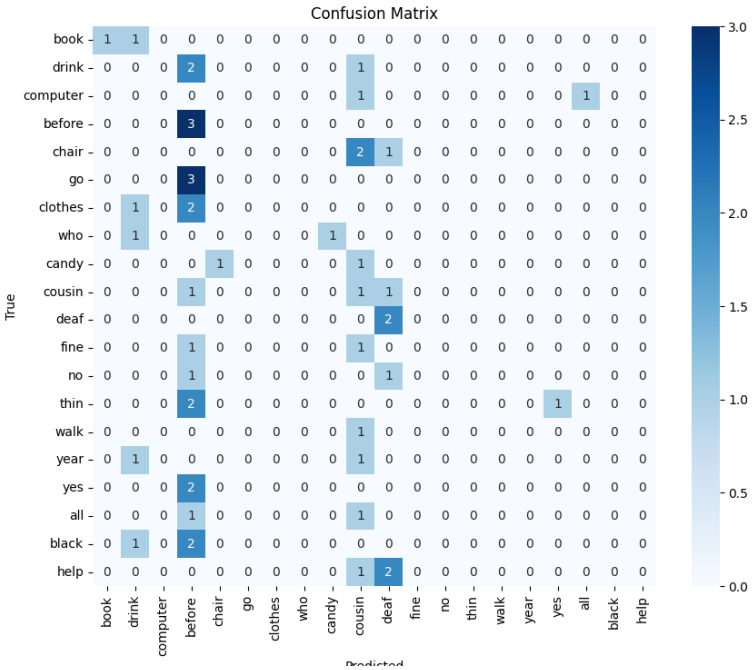

Figure 14: The confusion matrix for the second experiment's 3DCNN, employing the "ALL" data processing technique.

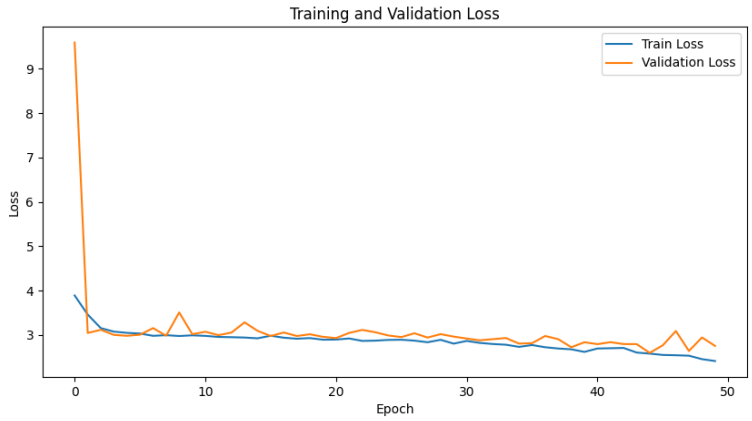

Figure 15: The loss function graph obtained from training the 3DCNN in the second experiment, employing the "ALL" data processing technique.

A.3 SlowFast neural network with BODY AND HANDS data processing

Figure 16 portrays the confusion matrix for the SlowFast neural network of the second experiment using the "BODY AND HANDS" data processing technique. Additionally, Figure 17 depicts the model's loss function during training. This particular model attained an accuracy level of 63.27%.

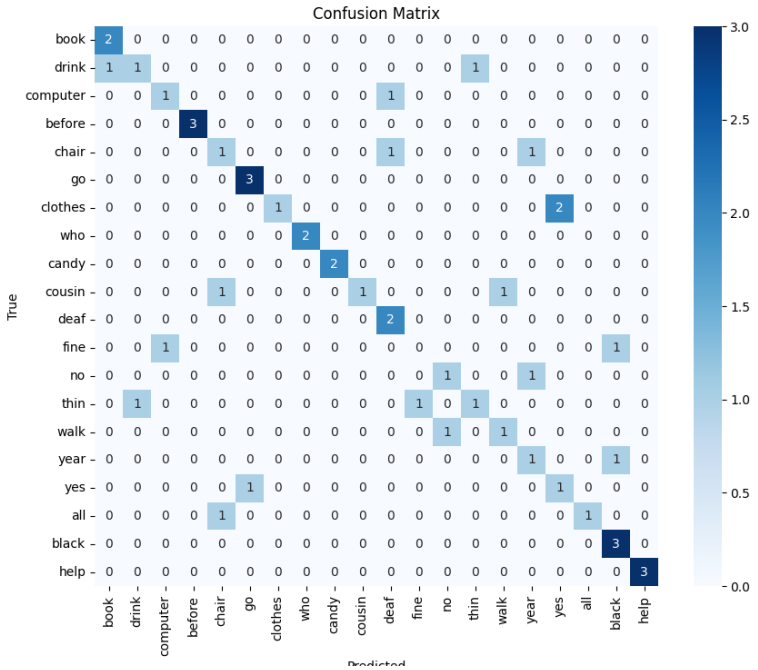

Figure 16: The confusion matrix for the second experiment's SlowFast neural network, employing the "BODY AND HANDS" data processing technique.

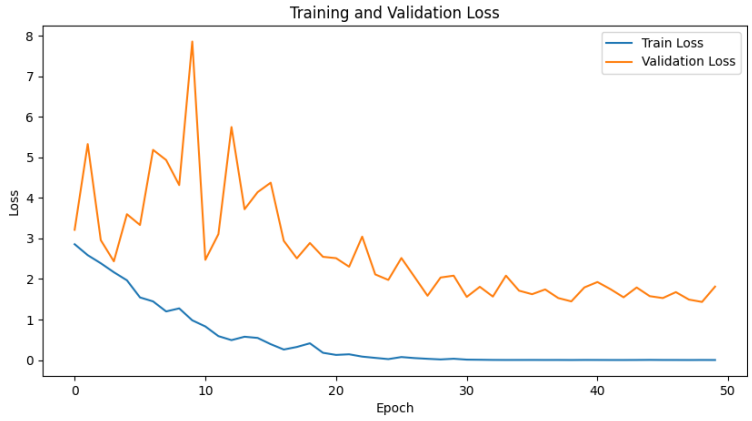

Figure 17: The loss function graph obtained from training the SlowFast neural network in the second experiment, employing the "BODY AND HANDS" data processing technique.

A.4 3DCNN with BODY AND HANDS data processing

Figure 18 shows the confusion matrix for the 3DCNN of the second experiment using the "BODY AND HANDS" data processing technique. Additionally, Figure 19 shows the loss function of the model during training. This particular model attained an accuracy level of 18.37%.

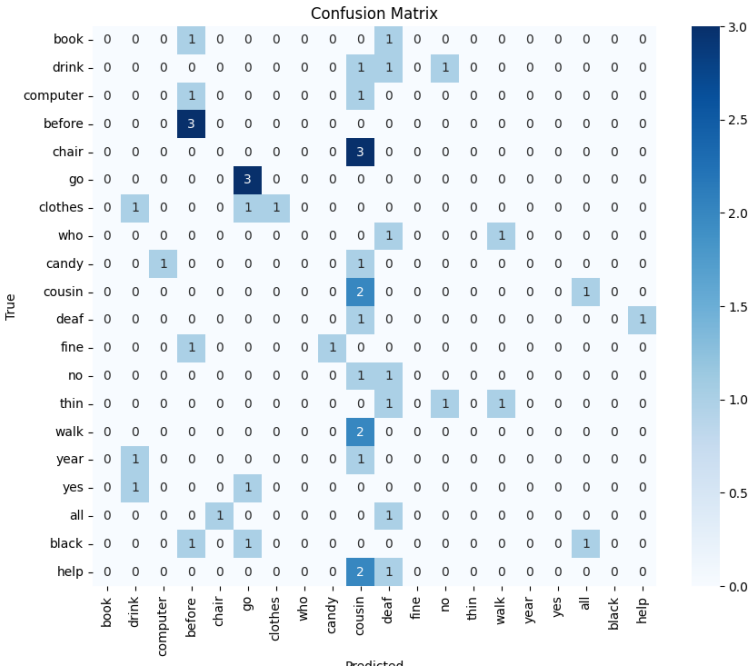

Figure 18: The confusion matrix for the second experiment's 3DCNN, employing the "BODY AND HANDS" data processing technique.

Figure 19: The loss function graph obtained from training the 3DCNN in the second experiment, employing the "BODY AND HANDS" data processing technique.

A.5 SlowFast neural network with FACE AND HANDS data processing

Figure 20 portrays the confusion matrix for the SlowFast neural network of the second experiment using the "FACE AND HANDS" data processing technique. Additionally, Figure 21 depicts the model's loss function during training. This particular model attained an accuracy level of 62.50%.

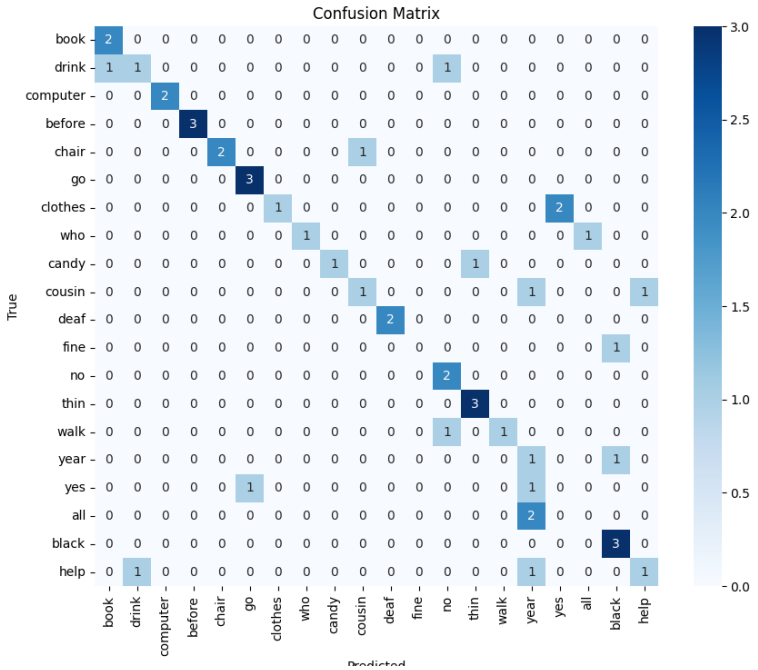

Figure 20: The confusion matrix for the second experiment's SlowFast neural network, employing the "FACE AND HANDS" data processing technique.

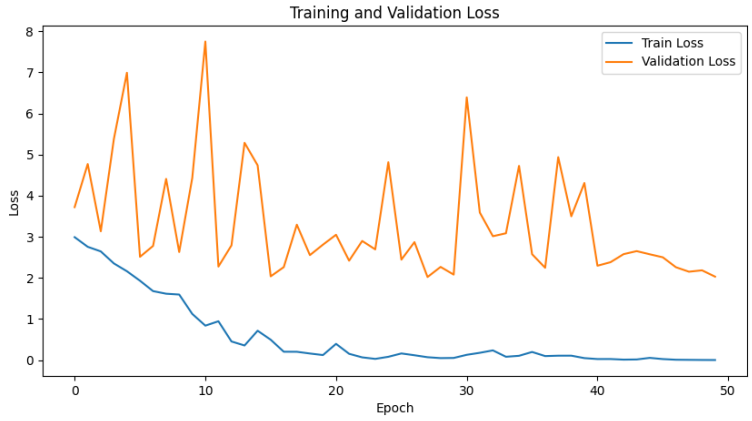

Figure 21: The loss function graph obtained from training the SlowFast neural network in the second experiment, employing the "FACE AND HANDS" data processing technique.

A.6 3DCNN with FACE AND HANDS data processing

Figure 22 shows the confusion matrix for the 3DCNN of the second experiment using the "FACE AND HANDS" data processing technique. Additionally, Figure 23 shows the loss function of the model during training. This particular model attained an accuracy level of 14.58%.

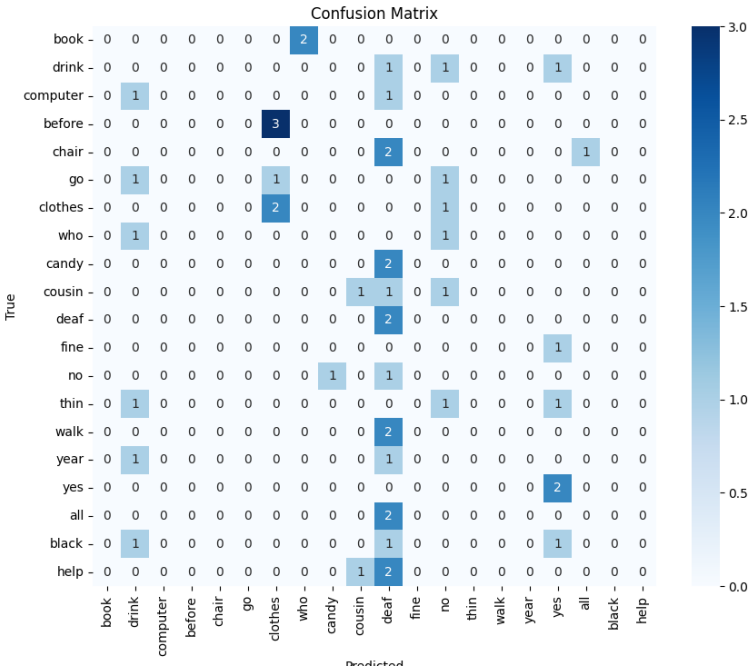

Figure 22: The confusion matrix for the second experiment's 3DCNN, employing the "FACE AND HANDS" data processing technique.

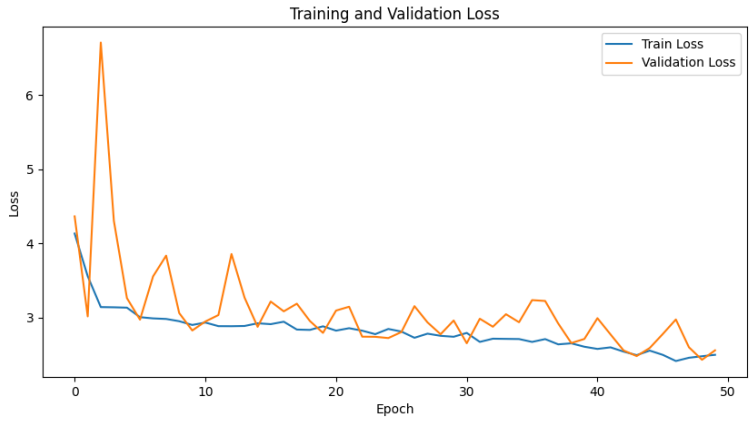

Figure 23: The loss function graph obtained from training the 3DCNN in the second experiment, employing the "FACE AND HANDS" data processing technique.

A.7 SlowFast neural network with HANDS data processing

Figure 24 portrays the confusion matrix for the SlowFast neural network of the second experiment using the "HANDS" data processing technique. Additionally, Figure 25 depicts the model's loss function during training. This particular model attained an accuracy level of 53.06%.

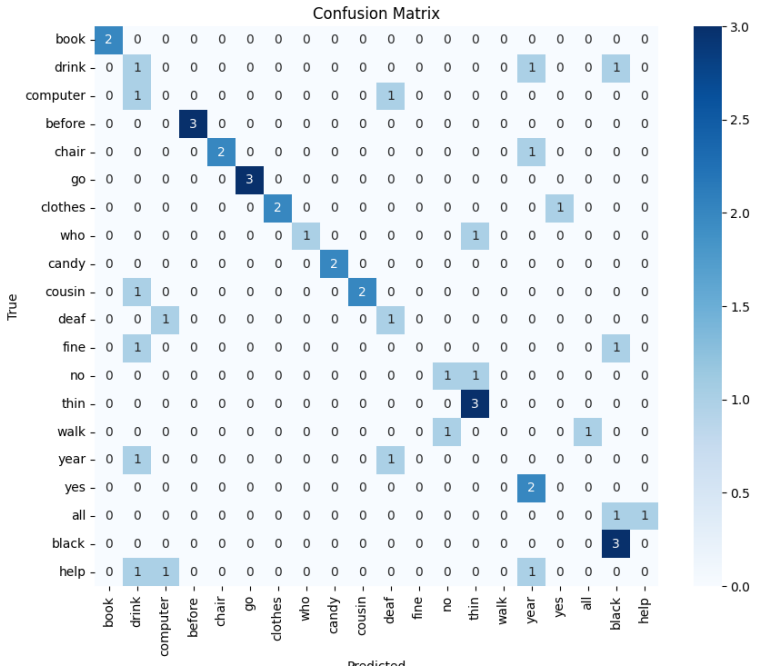

Figure 24: The confusion matrix for the second experiment's SlowFast neural network, employing the "HANDS" data processing technique.

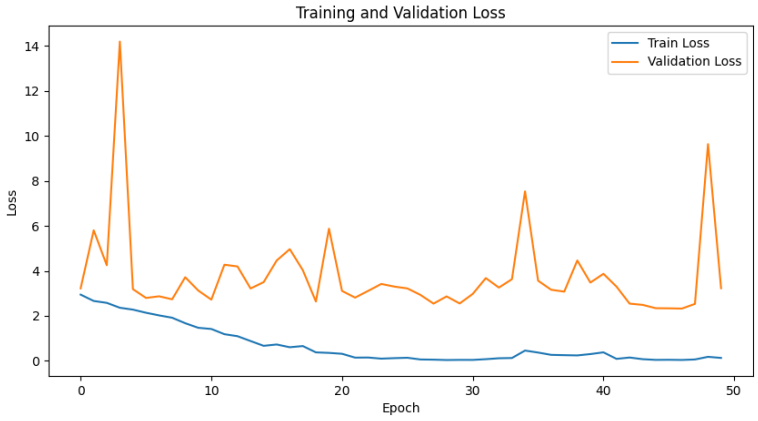

Figure 25: The loss function graph obtained from training the SlowFast neural network in the second experiment, employing the "HANDS" data processing technique.

A.8 3DCNN with HANDS data processing

Figure 26 shows the confusion matrix for the 3DCNN of the second experiment using the "HANDS" data processing technique. Additionally, Figure 27 shows the loss function of the model during training. This particular model attained an accuracy level of 26.53%.

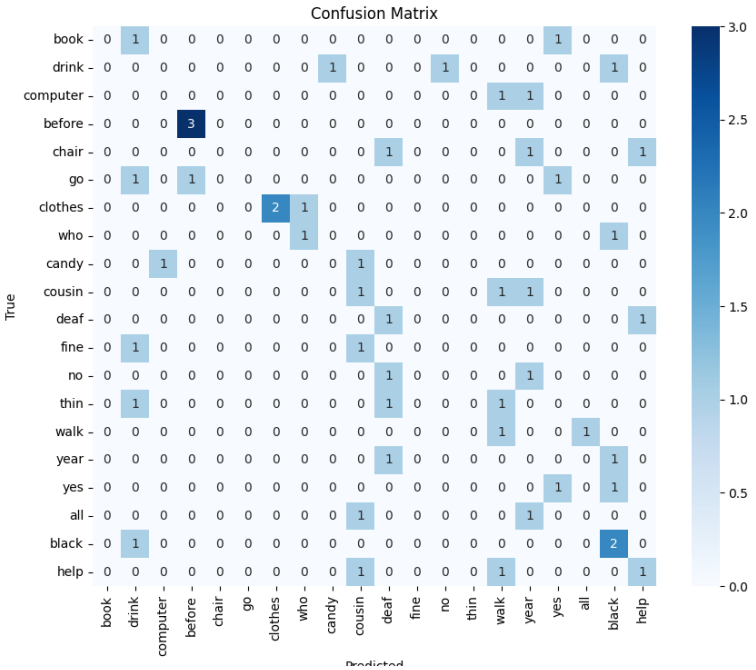

Figure 26: The confusion matrix for the second experiment's 3DCNN, employing the "HANDS" data processing technique.

Figure 27: The loss function graph obtained from training the 3DCNN in the second experiment, employing the "HANDS" data processing technique.

B Third experiment

B.1 SlowFast neural network using the WLASL dataset for training

Figure 28 portrays the confusion matrix for the SlowFast neural network of the third experiment using the WLASL dataset for training and the MSASL dataset for testing. Additionally, Figure 29 depicts the model's loss function during training. This particular model attained an accuracy level of 44.44%.

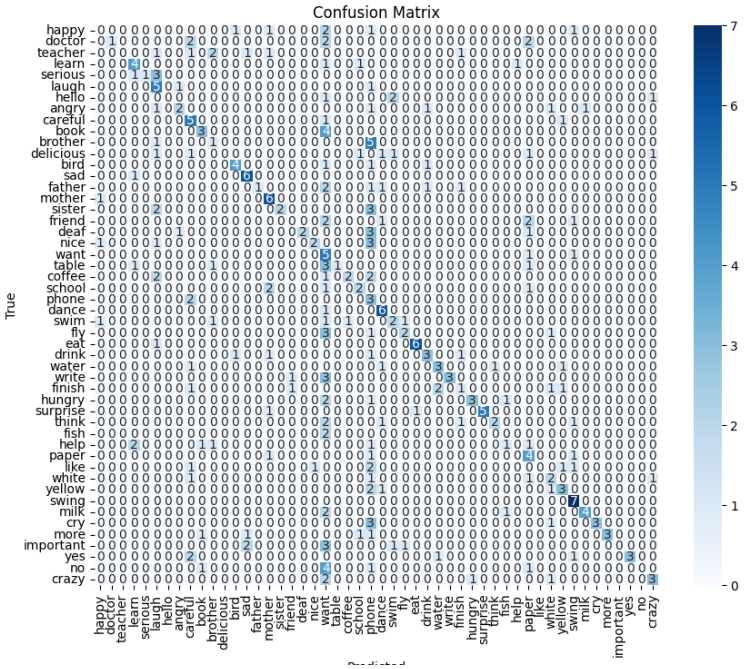

Figure 28: The confusion matrix for the third experiment's SlowFast neural network, using the WLASL dataset for training.

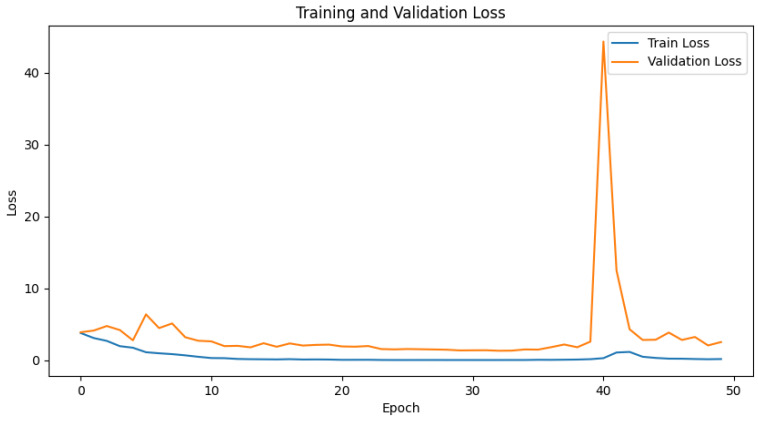

Figure 29: The loss function graph obtained from training the SlowFast neural network in the third experiment using the WLASL dataset.

B.2 3DCNN using the WLASL dataset for training

Figure 30 portrays the confusion matrix for the 3DCNN of the third experiment using the WLASL dataset for training and the MSASL dataset for testing. Additionally, Figure 31 depicts the model's loss function during training. This particular model attained an accuracy level of 44.44%.

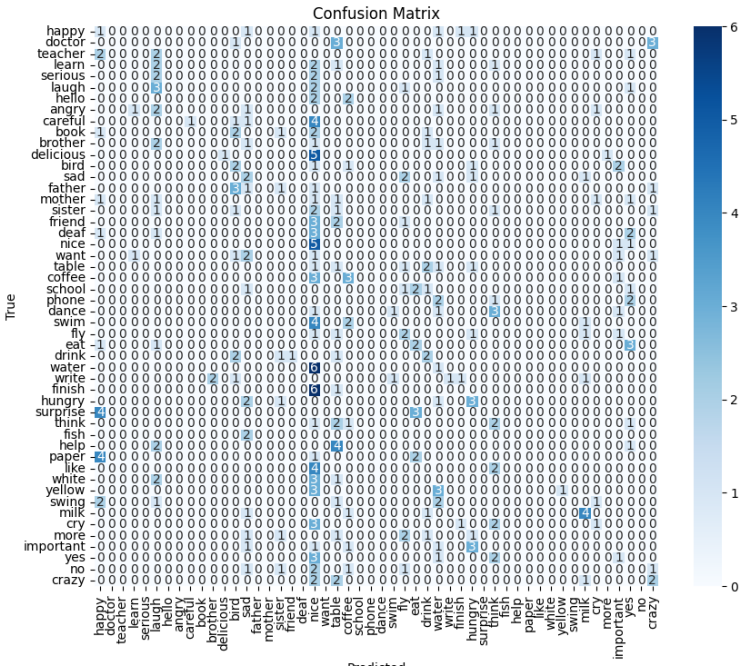

Figure 30: The confusion matrix for the third experiment's 3DCNN, using the WLASL dataset for training.

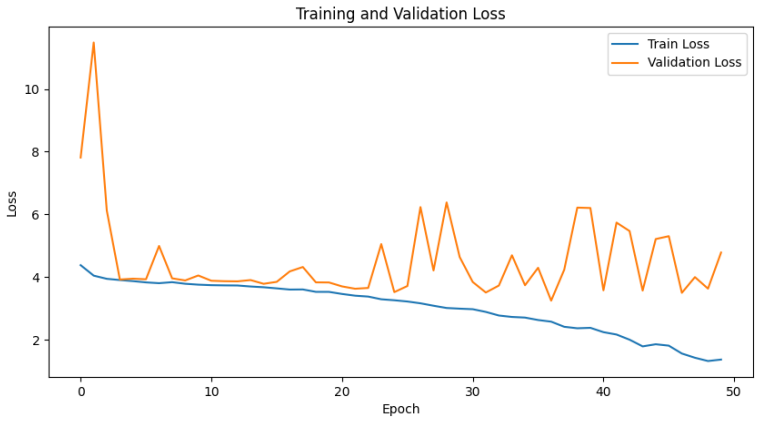

Figure 31: The loss function graph obtained from training the 3DCNN in the third experiment using the WLASL dataset.

B.3 SlowFast neural network using mixed precision and the WLASL dataset for training

Figure 32 portrays the confusion matrix for the SlowFast neural network of the third experiment using the WLASL dataset for training and the MSASL dataset for testing. The batch size was increased to 10, using the mixed precision technique. Additionally, Figure 33 depicts the model's loss function during training. This particular model achieved an accuracy level of 37.84%.

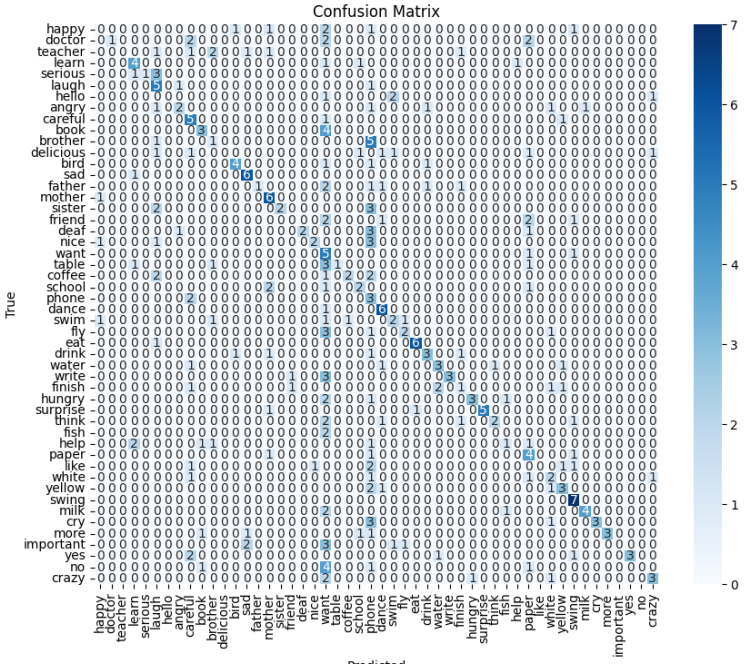

Figure 32: The confusion matrix for the third experiment's SlowFast neural network, using the WLASL dataset for training.

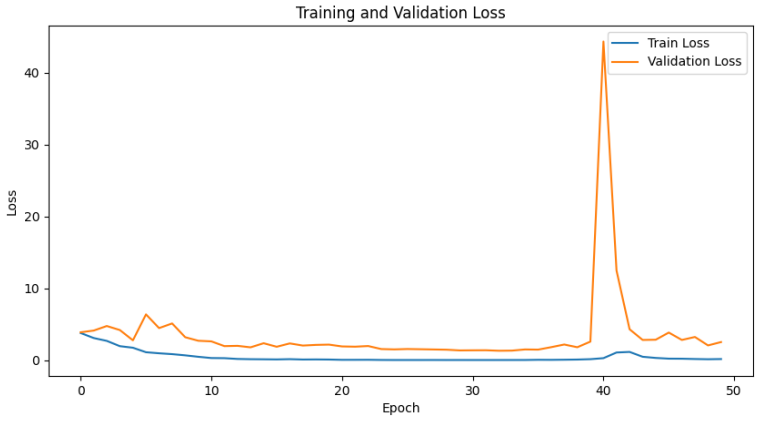

Figure 33: The loss function graph obtained from training the SlowFast neural network in the third experiment using the WLASL dataset.

B.4 3DCNN using mixed precision and the WLASL dataset for training

Figure 34 portrays the confusion matrix for the 3DCNN of the third experiment using the WLASL dataset for training and the MSASL dataset for testing. The batch size was increased to 10, using the mixed precision technique. Additionally, Figure 35 depicts the model's loss function during training. This particular model achieved an accuracy level of 12.01%.

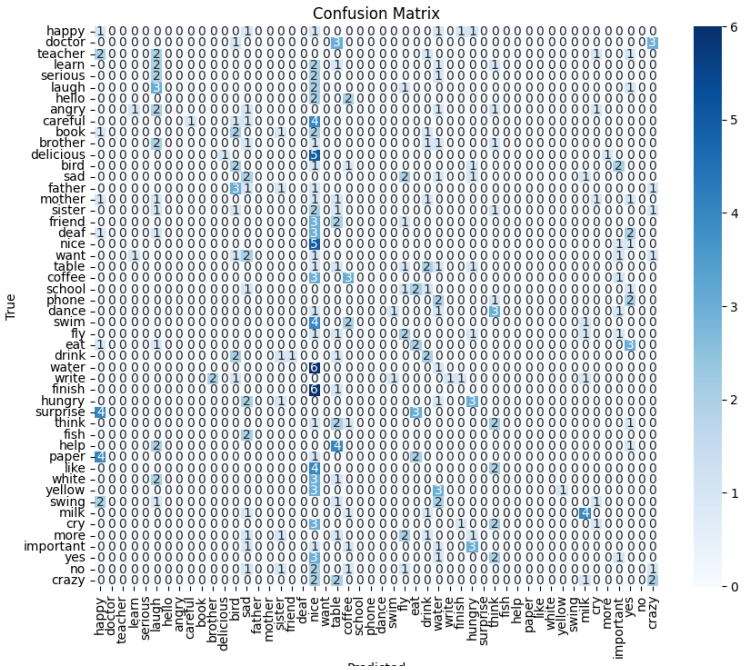

Figure 34: The confusion matrix for the third experiment's 3DCNN, using the WLASL dataset for training.

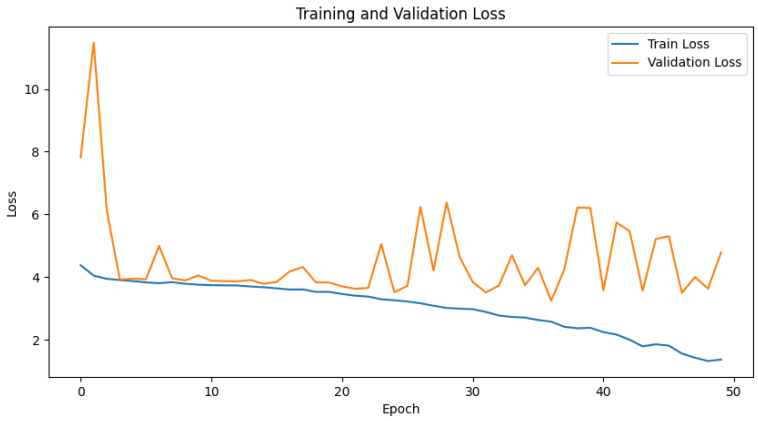

Figure 35: The loss function graph obtained from training the 3DCNN in the third experiment using the WLASL dataset.

B.5 Comparison of the different training and testing datasets

B.5.1 SlowFast neural network using the WLASL dataset for training

Figure 36 portrays the confusion matrix for the SlowFast neural network of the third experiment using the WLASL dataset for training. Additionally, Figure 33 depicts the model's loss function during training. This particular model achieved an accuracy level of 32.53%.

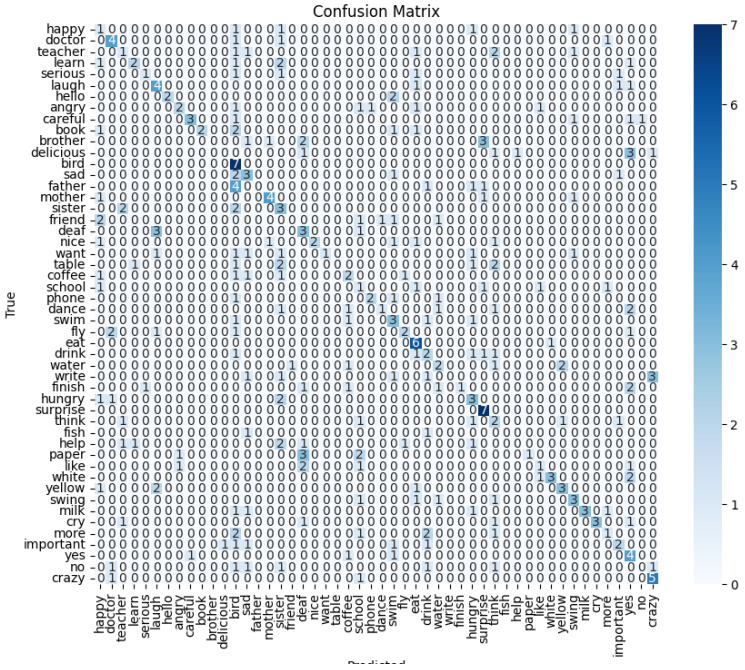

Figure 36: The confusion matrix for the third experiment's SlowFast neural network, using the WLASL dataset for training.

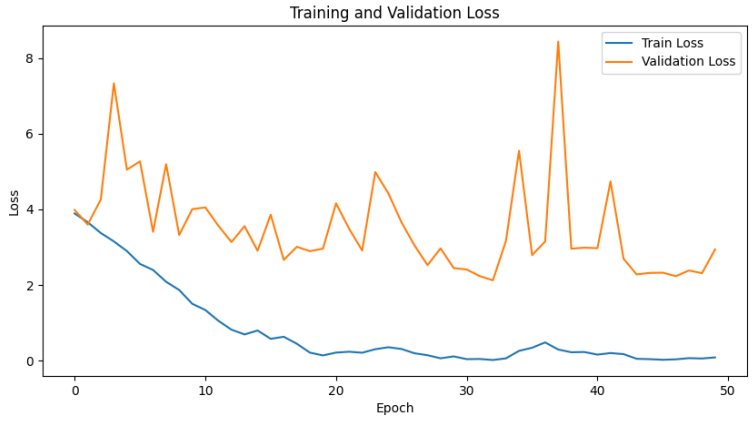

Figure 37: The loss function graph obtained from training the SlowFast neural network in the third experiment using the WLASL dataset.

B.5.2 Facebook 3DCNN model using the WLASL dataset for training

Figure 38 portrays the confusion matrix for the 3DCNN model from Facebook of the third experiment using the WLASL dataset for training. Additionally, Figure 35 depicts the model's loss function during training. This particular model achieved an accuracy level of 20.94%.

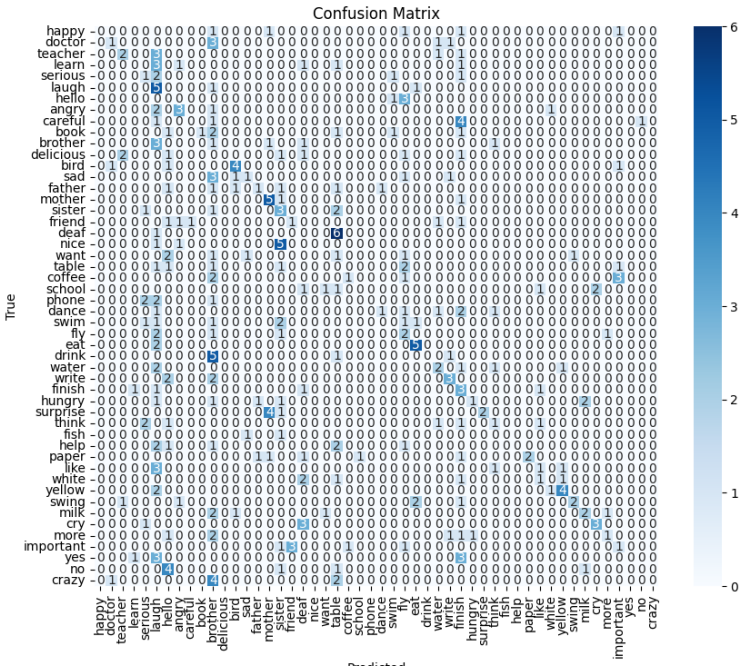

Figure 38: The confusion matrix for the third experiment's 3DCNN pre-trained by Facebook, using the WLASL dataset for training.

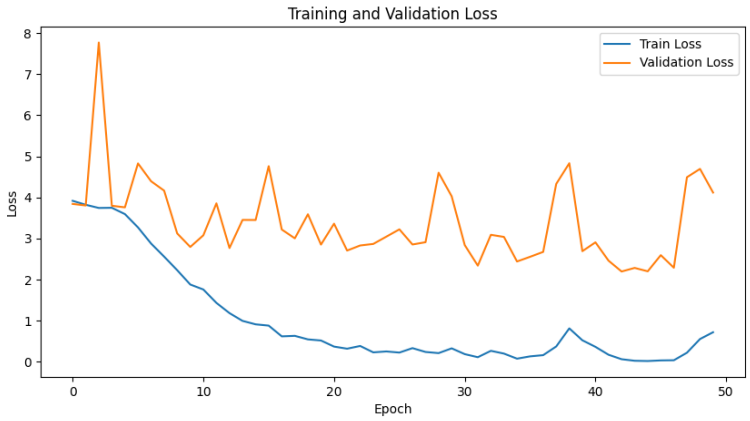

Figure 39: The loss function graph obtained from training the 3DCNN pre-trained by Facebook in the third experiment, using the WLASL dataset.

B.5.3 SlowFast neural network using the MSASL dataset for training

Figure 40 portrays the confusion matrix for the SlowFast neural network of the third experiment using the MSASL dataset for training. Additionally, Figure 41 depicts the model's loss function during training. This particular model achieved an accuracy level of 32.53%.

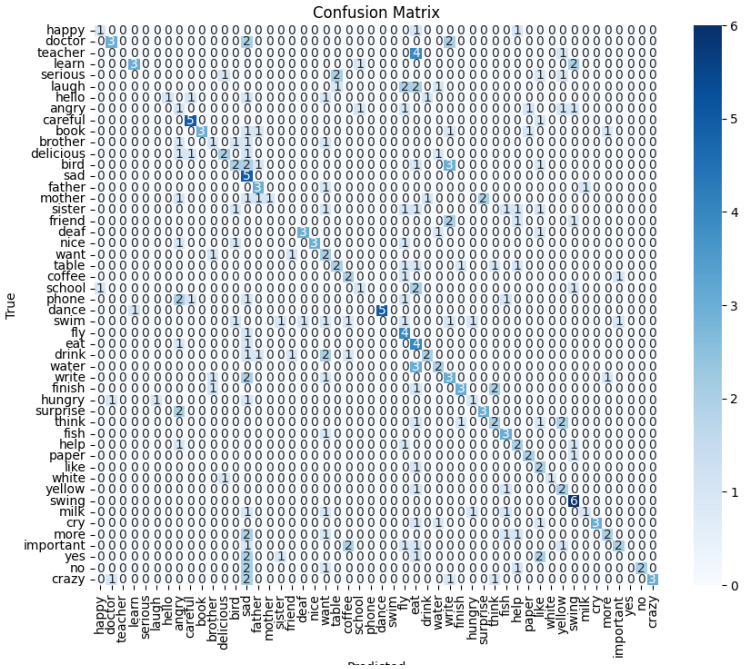

Figure 40: The confusion matrix for the third experiment's SlowFast neural network, using the MSASL dataset for training.

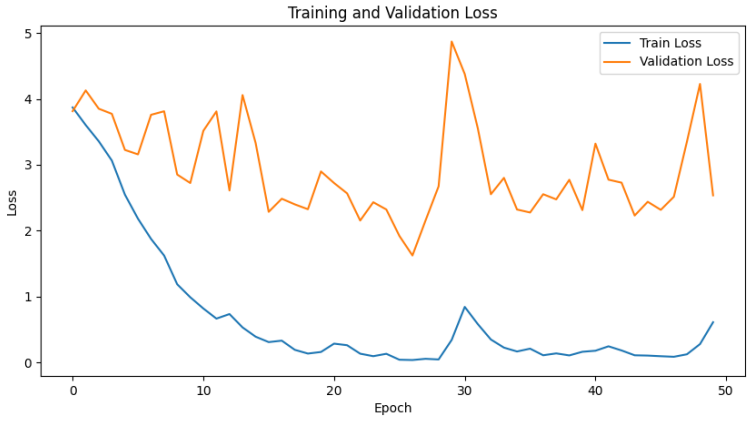

Figure 41: The loss function graph obtained from training the SlowFast neural network in the third experiment using the MSASL dataset.

B.5.4 Facebook 3DCNN model using the MSASL dataset for training

Figure 42 portrays the confusion matrix for the 3DCNN model from Facebook of the third experiment using the MSASL dataset for training. Additionally, Figure 43 depicts the model's loss function during training. This particular model achieved an accuracy level of 20.94%.

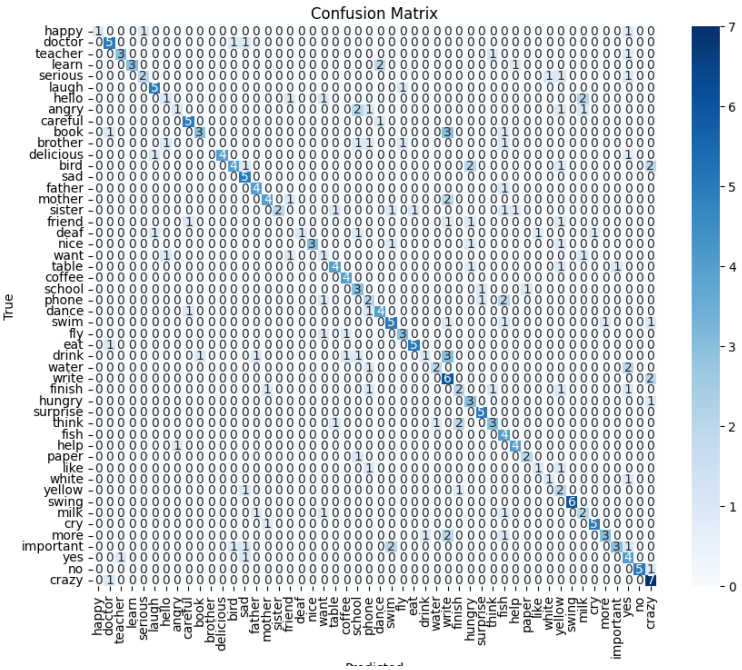

Figure 42: The confusion matrix for the third experiment's 3DCNN pre-trained by Facebook, using the MSASL dataset for training.

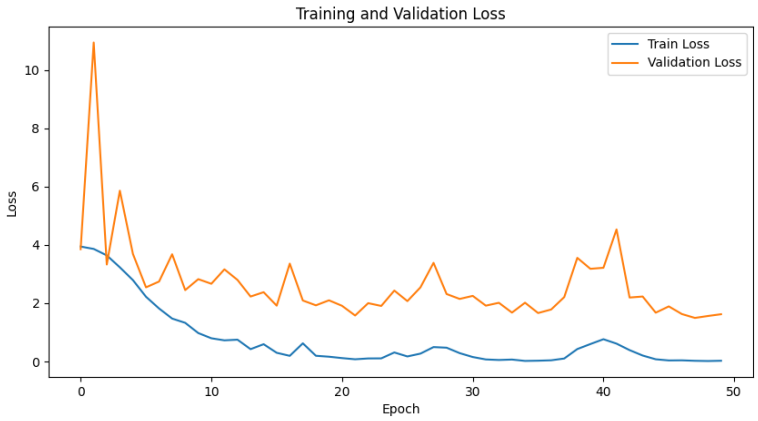

Figure 43: The loss function graph obtained from training the 3DCNN pre-trained by Facebook in the third experiment, using the MSASL dataset.

B.5.5 SlowFast neural network using both datasets for training

Figure 44 portrays the confusion matrix for the SlowFast neural network of the third experiment using both datasets for training. Additionally, Figure 45 depicts the model's loss function during training. This particular model achieved an accuracy level of 32.53%.

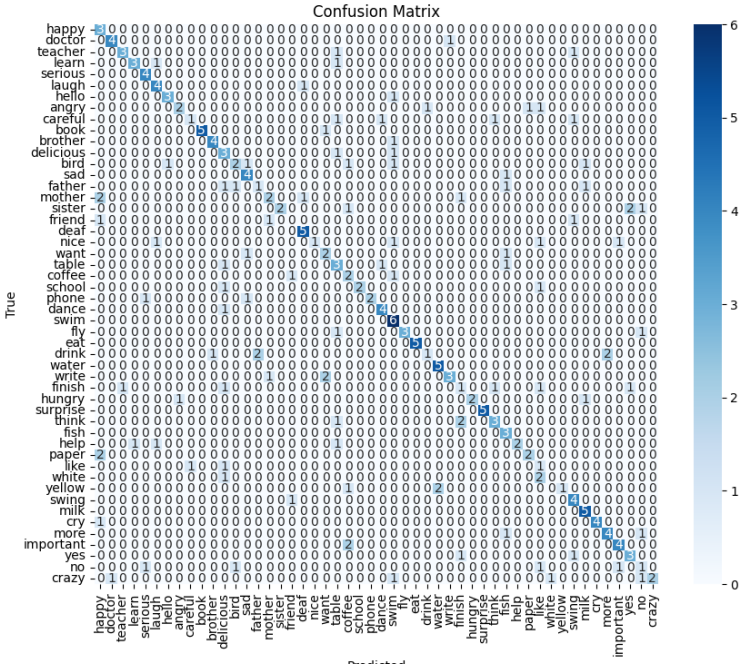

Figure 44: The confusion matrix for the third experiment's SlowFast neural network, using both datasets for training.

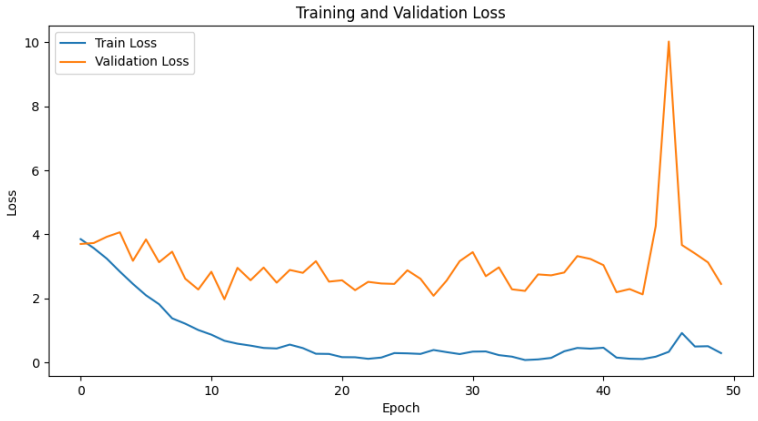

Figure 45: The loss function graph obtained from training the SlowFast neural network in the third experiment using both datasets.

B.5.6 Facebook 3DCNN model using both datasets for training

Figure 46 portrays the confusion matrix for the 3DCNN model from Facebook of the third experiment using both datasets for training. Additionally, Figure 47 depicts the model's loss function during training. This particular model achieved an accuracy level of 20.94%.

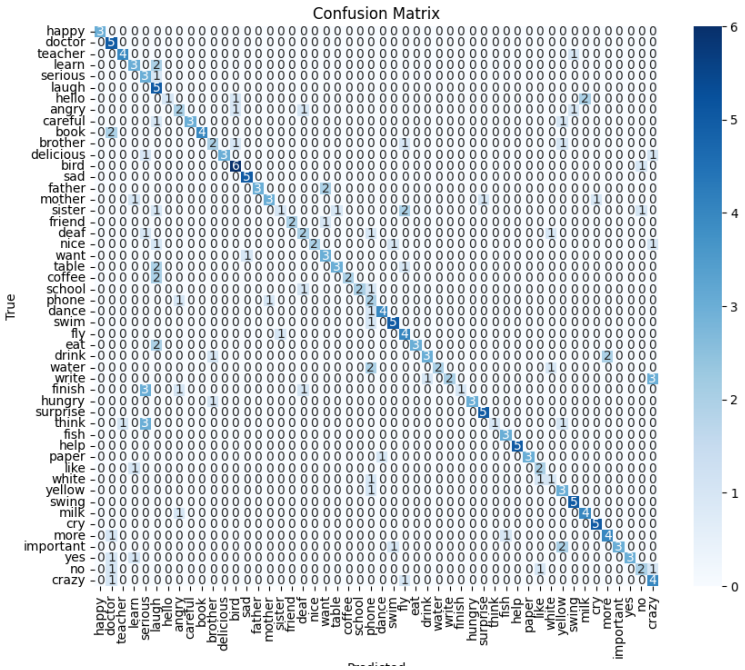

Figure 46: The confusion matrix for the third experiment's 3DCNN pre-trained by Facebook, using both datasets for training.

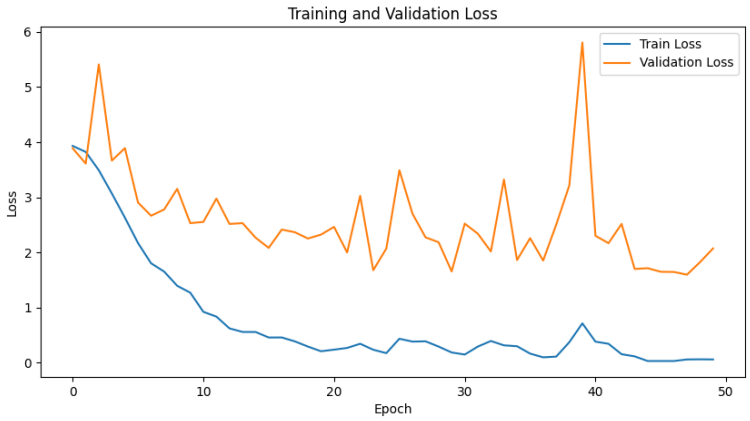

Figure 47: The loss function graph obtained from training the 3DCNN pre-trained by Facebook in the third experiment, using both datasets.

B.6 Comparison of the different training and testing datasets after removing duplicated samples

B.6.1 SlowFast neural network using the WLASL dataset for training

Figure 48 portrays the confusion matrix for the SlowFast neural network of the third experiment using the WLASL datasets for training, after having removed the duplicated samples from both datasets. Additionally, Figure 49 depicts the model's loss function during training. This particular model achieved an accuracy level of 30.97%.

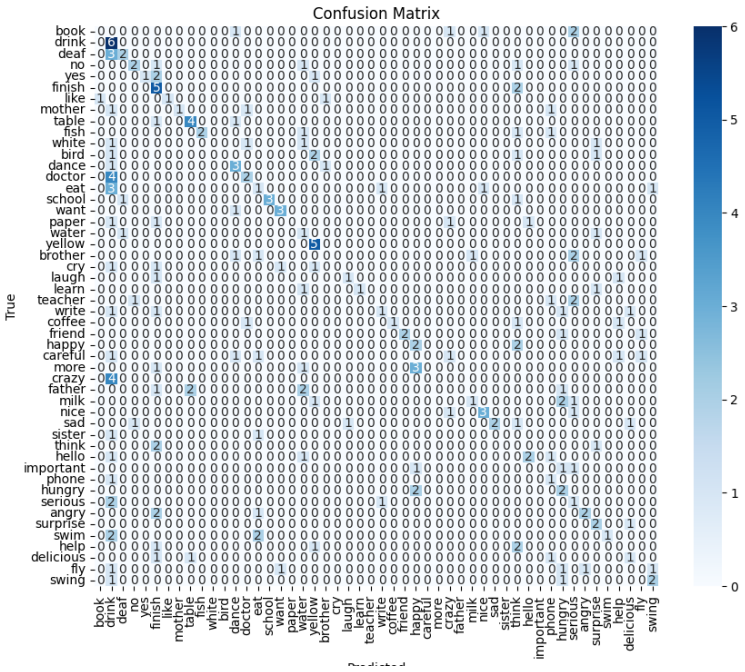

Figure 48: The confusion matrix for the third experiment's SlowFast neural network, using WLASL datasets for training, after removing duplicated samples.

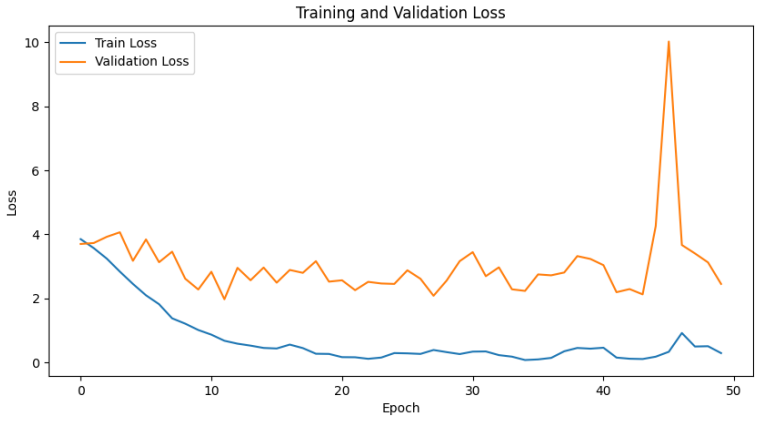

Figure 49: The loss function graph obtained from training the SlowFast neural network in the third experiment using WLASL dataset for training, after removing duplicated samples.

B.6.2 3DCNN using the WLASL dataset for training

Figure 50 portrays the confusion matrix for the 3DCNN of the third experiment using the WLASL datasets for training, after having removed the duplicated samples from both datasets. Additionally, Figure 51 depicts the model's loss function during training. This particular model achieved an accuracy level of 45.58%.

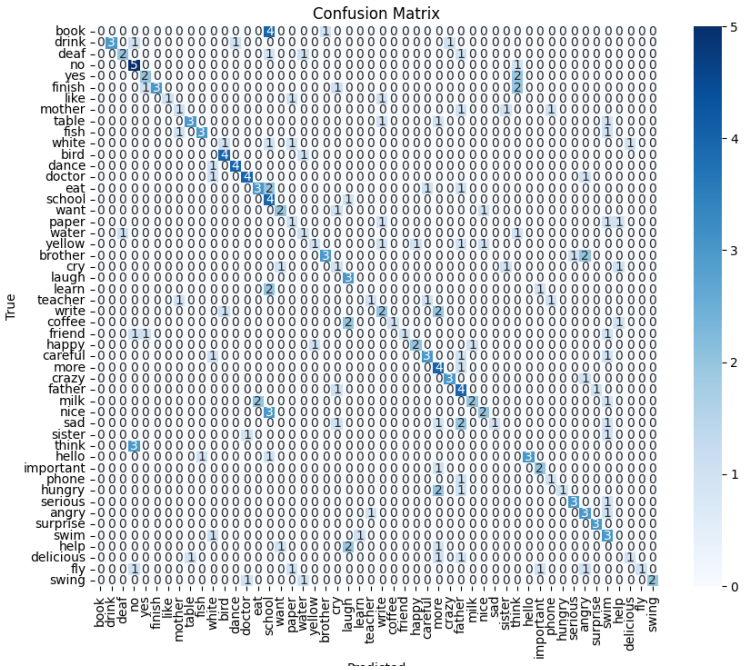

Figure 50: The confusion matrix for the third experiment's 3DCNN, using MSASL datasets for training, after removing duplicated samples.

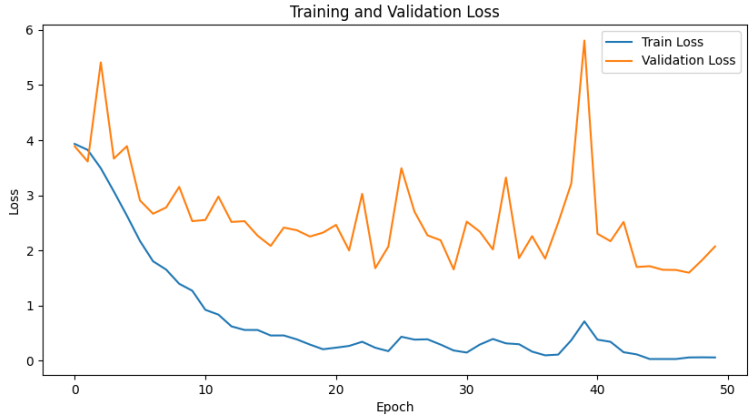

Figure 51: The loss function graph obtained from training the 3DCNN in the third experiment using MSASL dataset for training, after removing duplicated samples.

B.6.3 SlowFast neural network using the MSASL dataset for training

Figure 52 portrays the confusion matrix for the SlowFast neural network of the third experiment using the MSASL datasets for training, after having removed the duplicated samples from both datasets. Additionally, Figure 53 depicts the model's loss function during training. This particular model achieved an accuracy level of29.20%.

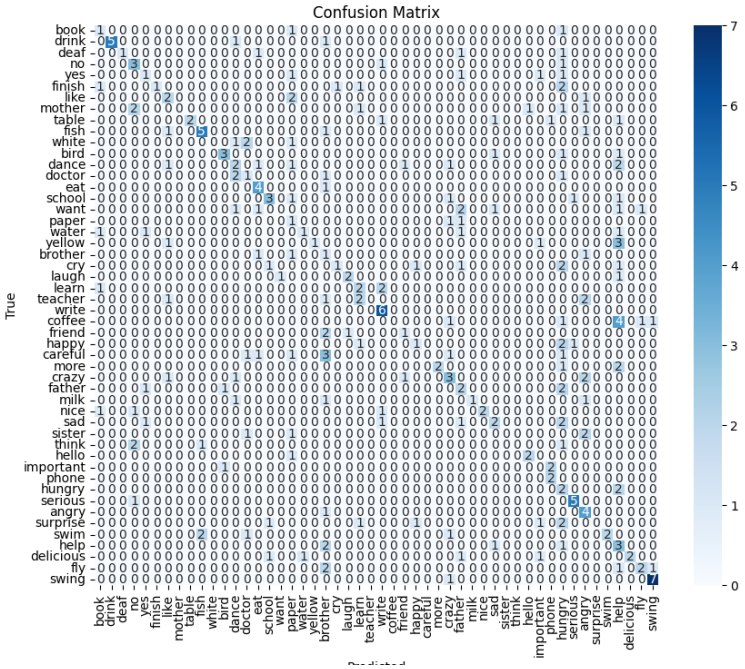

Figure 52: The confusion matrix for the third experiment's SlowFast neural network, using MSASL datasets for training, after removing duplicated samples.

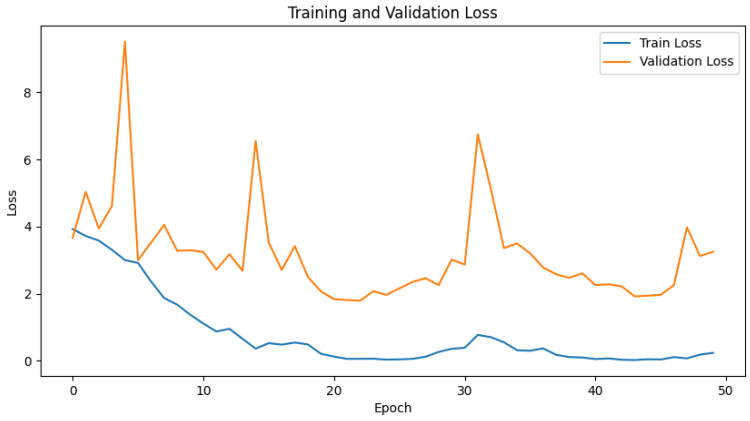

Figure 53: The loss function graph obtained from training the SlowFast neural network in the third experiment using MSASL dataset for training, after removing duplicated samples.

B.6.4 3DCNN using the MSASL dataset for training

Figure 50 portrays the confusion matrix for the 3DCNN of the third experiment using the MSASL datasets for training, after having removed the duplicated samples from both datasets. Additionally, Figure 55 depicts the model's loss function during training. This particular model achieved an accuracy level of 34.31%.

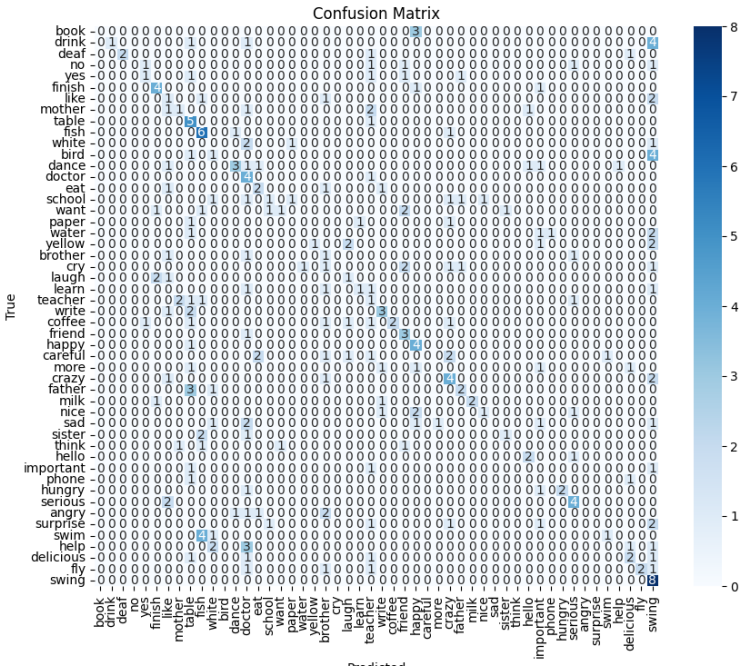

Figure 54: The confusion matrix for the third experiment's 3DCNN, using MSASL datasets for training, after removing duplicated samples.

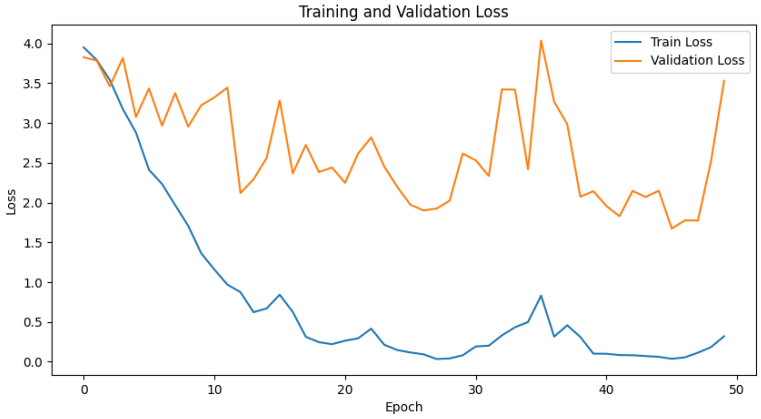

Figure 55: The loss function graph obtained from training the 3DCNN in the third experiment using MSASL dataset for training, after removing duplicated samples.

B.6.5 SlowFast neural network using both datasets for training

Figure 56 portrays the confusion matrix for the SlowFast neural network of the third experiment using the both datasets for training, after having removed the duplicated samples from both datasets. Additionally, Figure 57 depicts the model's loss function during training. This particular model achieved an accuracy level of 47.62%.

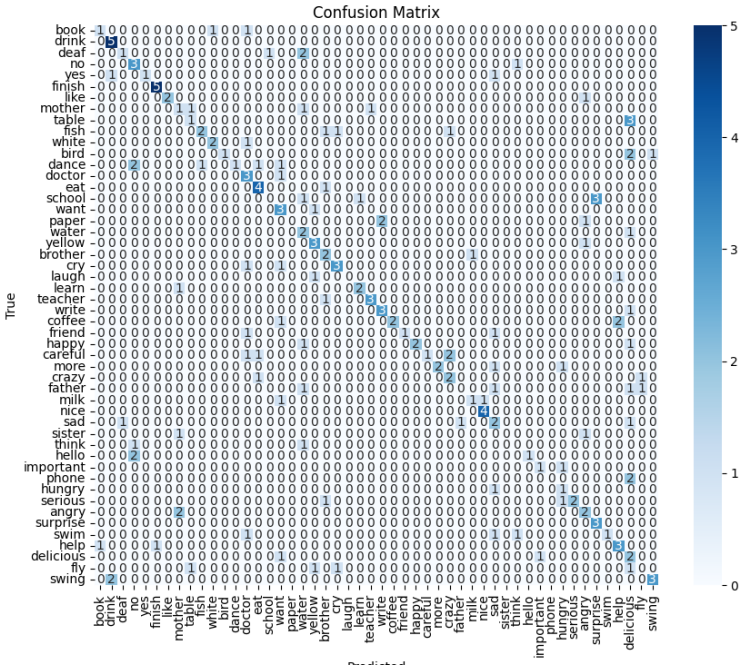

Figure 56: The confusion matrix for the third experiment's SlowFast neural network, using both datasets for training, after removing duplicated samples.

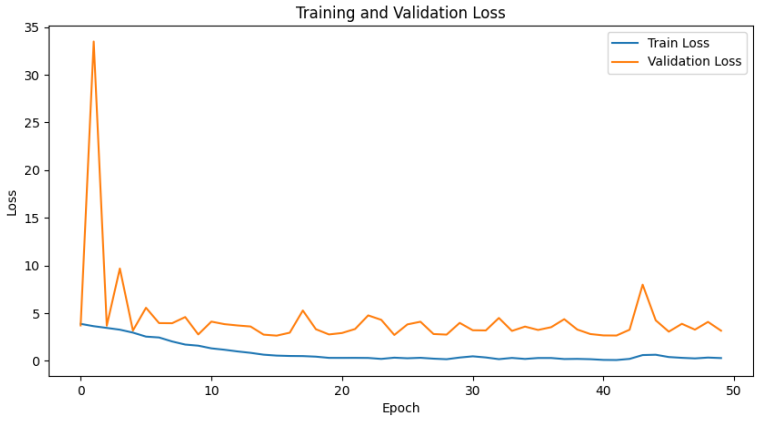

Figure 57: The loss function graph obtained from training the SlowFast neural network in the third experiment using both datasets for training, after removing duplicated samples.

B.6.6 3DCNN using both datasets for training

Figure 58 portrays the confusion matrix for the 3DCNN of the third experiment using the both datasets for training, after having removed the duplicated samples from both datasets. Additionally, Figure 47 depicts the model's loss function during training. This particular model achieved an accuracy level of 57.67%.

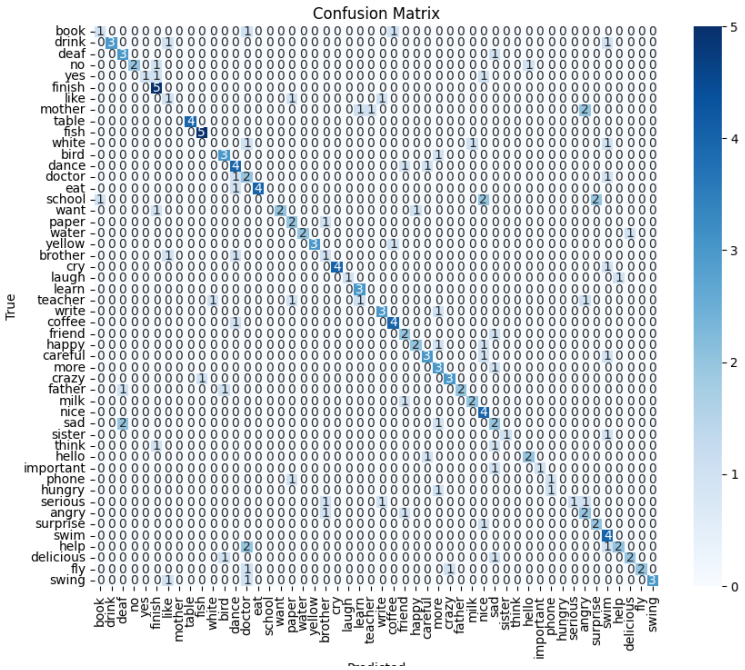

Figure 58: The confusion matrix for the third experiment's 3DCNN, using both datasets for training, after removing duplicated samples.

Figure 59: The loss function graph obtained from training the 3DCNN in the third experiment using both datasets for training, after removing duplicated samples.

C Fourth experiment

C.1 Training over the 50 labels

C.1.1 SlowFast neural network using the FACE AND HANDS data processing

Figure 60 portrays the confusion matrix for the 3DCNN of the fourth experiment trained with the 50 labels of Table 4 and applying to the input data the FACE AND HANDS data processing technique explained on Section 3.2.2. Additionally, Figure 61 depicts the model's loss function during training. This particular model achieved an accuracy level of 50%.

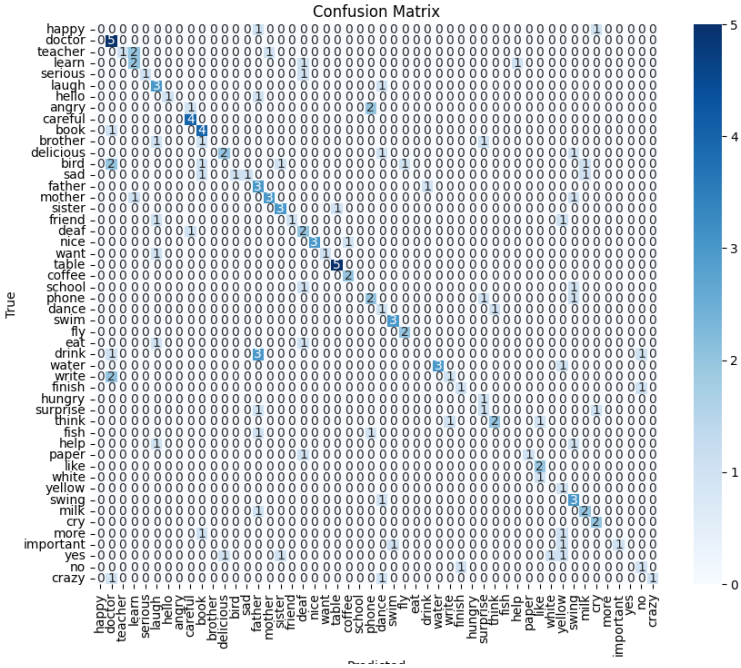

Figure 60: The confusion matrix for the fourth experiment's Slow Fast Neural Network, using all labels and the FACE AND HANDS data processing technique.

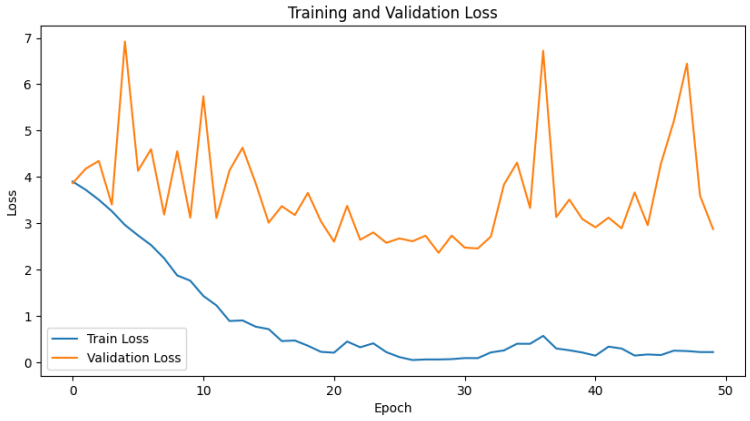

Figure 61: The loss function graph obtained from training the Slow Fast Neural Network in the fourth experiment using all labels and the FACE AND HANDS data processing technique.

C.1.2 3DCNN using the ALL data processing

Figure 62 portrays the confusion matrix for the 3DCNN of the fourth experiment trained with the 50 labels of Table 4 and applying to the input data the ALL data processing technique explained on Section 3.2.2. Additionally, Figure 63 depicts the model's loss function during training. This particular model achieved an accuracy level of 68.25%.

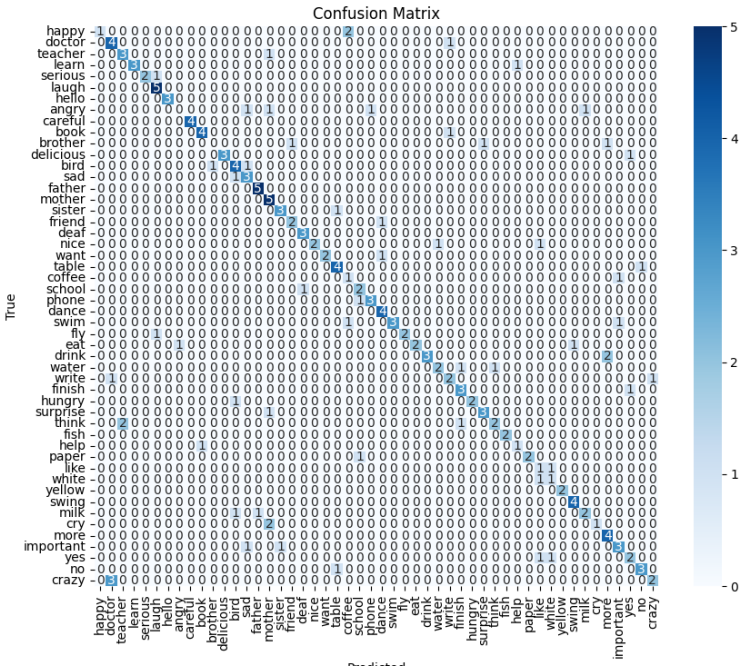

Figure 62: The confusion matrix for the fourth experiment's 3DCNN, using all labels and the ALL data processing technique.

Figure 63: The loss function graph obtained from training the 3DCNN in the fourth experiment using all labels and the ALL data processing technique.

C.2 Training over the hands gestures labels

C.2.1 SlowFast neural network using the ALL data processing

Figure 64 portrays the confusion matrix for the 3DCNN of the fourth experiment trained with the hands gestures labels of Table 4 and applying to the input data the ALL data processing technique explained on Section 3.2.2. Additionally, Figure 65 depicts the model's loss function during training. This particular model achieved an accuracy level of 76.64%.

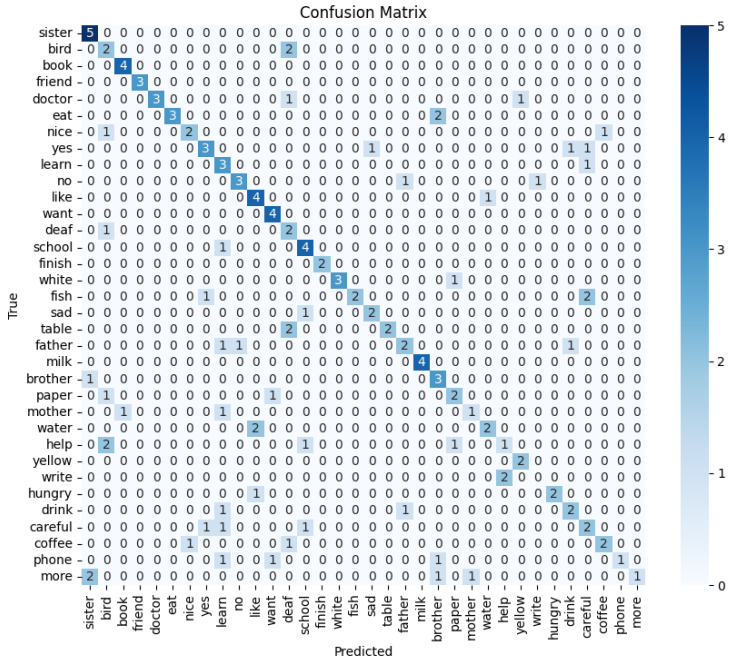

Figure 64: The confusion matrix for the fourth experiment's Slow Fast Neural Network, using all labels and the ALL data processing technique.

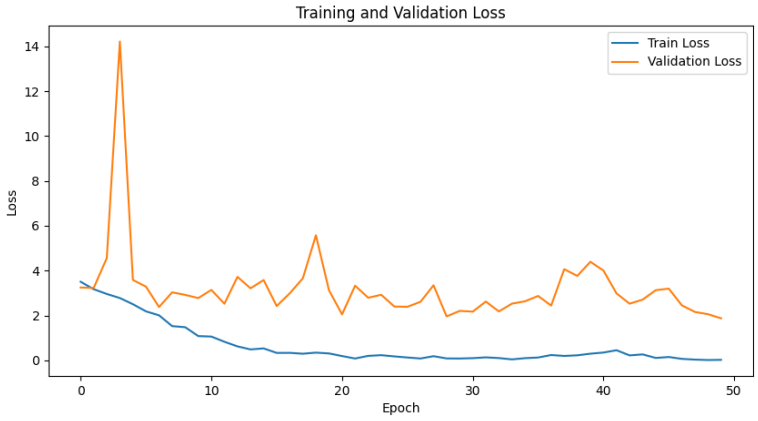

Figure 65: The loss function graph obtained from training the Slow Fast Neural Network in the fourth experiment using all labels and the ALL data processing technique.

C.2.2 3DCNN using the ALL data processing

Figure 66 portrays the confusion matrix for the 3DCNN of the fourth experiment trained with the hand gestures labels of Table 4 and applying to the input data the ALL data processing technique explained on Section 3.2.2. Additionally, Figure 67 depicts the model's loss function during training. This particular model achieved an accuracy level of 74.64%.

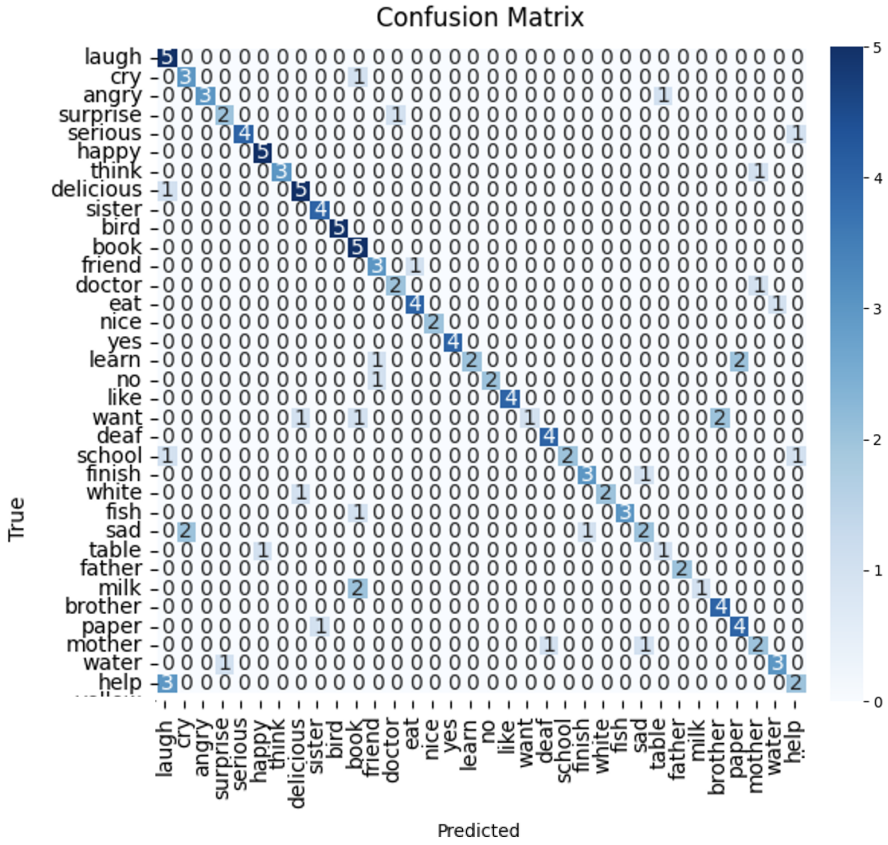

Figure 66: The confusion matrix for the fourth experiment's 3DCNN, using all labels and the ALL data processing technique.

Figure 67: The loss function graph obtained from training the 3DCNN in the fourth experiment using all labels and the ALL data processing technique.

C.3 Training over the face gestures labels

C.3.1 SlowFast neural network using the BODY AND HANDS data processing

Figure 68 portrays the confusion matrix for the 3DCNN of the fourth experiment trained with the face gestures labels of Table 4 and applying to the input data the BODY AND HANDS data processing technique explained on Section 3.2.2. Additionally, Figure 69 depicts the model's loss function during training. This particular model achieved an accuracy level of 70.37%.

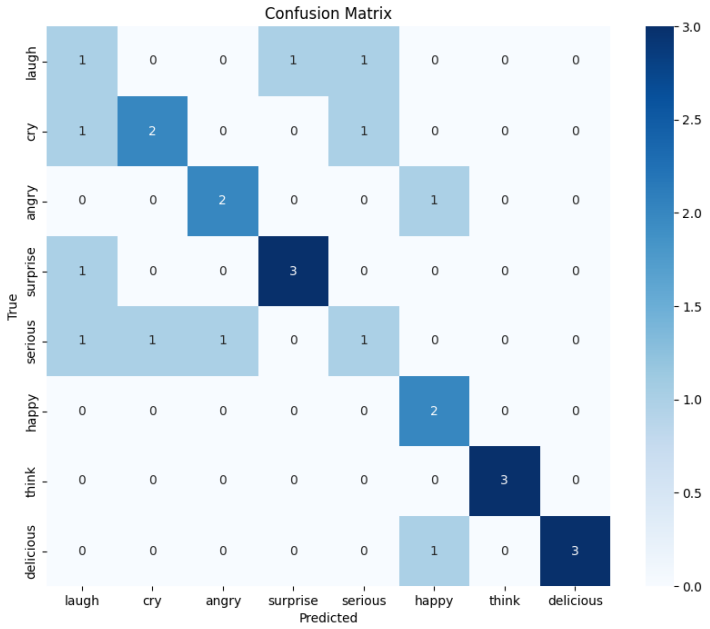

Figure 68: The confusion matrix for the fourth experiment's Slow Fast Neural Network, using all labels and the BODY AND HANDS data processing technique.

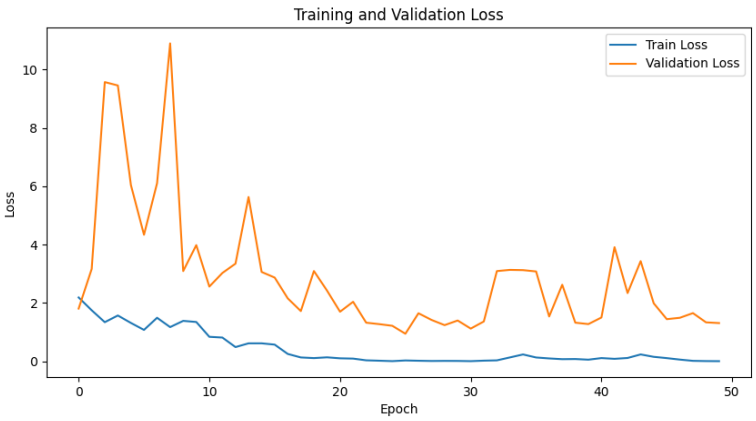

Figure 69: The loss function graph obtained from training the Slow Fast Neural Network in the fourth experiment using all labels and the BODY AND HANDS data processing technique.

C.3.2 3DCNN using the ALL data processing

Figure 70 portrays the confusion matrix for the 3DCNN of the fourth experiment trained with the face gestures labels of Table 4 and applying to the input data the ALL data processing technique explained on Section 3.2.2. Additionally, Figure 71 depicts the model's loss function during training. This particular model achieved an accuracy level of 62.96%.

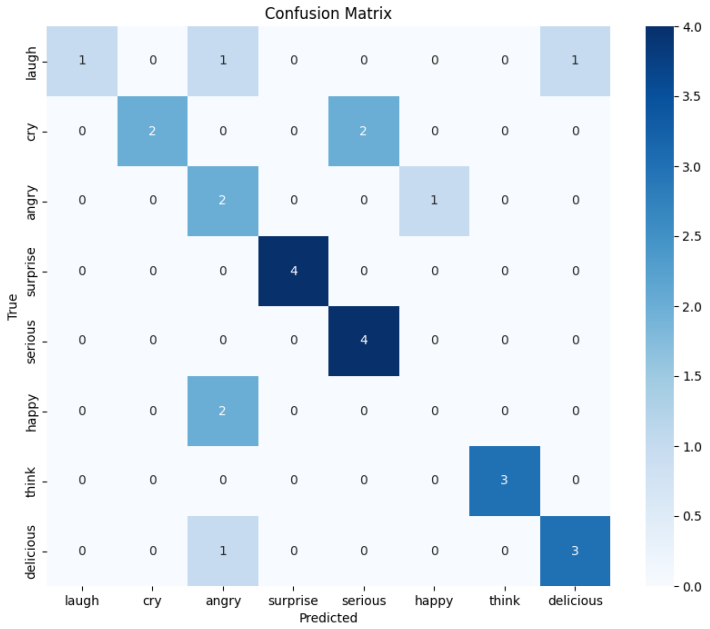

Figure 70: The confusion matrix for the fourth experiment's 3DCNN, using all labels and the ALL data processing technique.

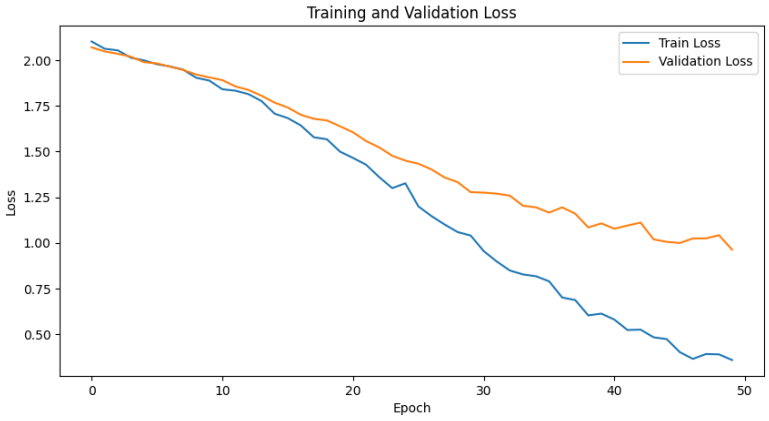

Figure 71: The loss function graph obtained from training the 3DCNN in the fourth experiment using all labels and the ALL data processing technique.

C.4 Training over the body gestures labels

C.4.1 SlowFast neural network using the BODY AND HANDS data processing

Figure 72 portrays the confusion matrix for the 3DCNN of the fourth experiment trained with the hands gestures labels of Table 4 and applying to the input data the BODY AND HANDS data processing technique explained on Section 3.2.2. Additionally, Figure 73 depicts the model's loss function during training. This particular model achieved an accuracy level of 63.04%.

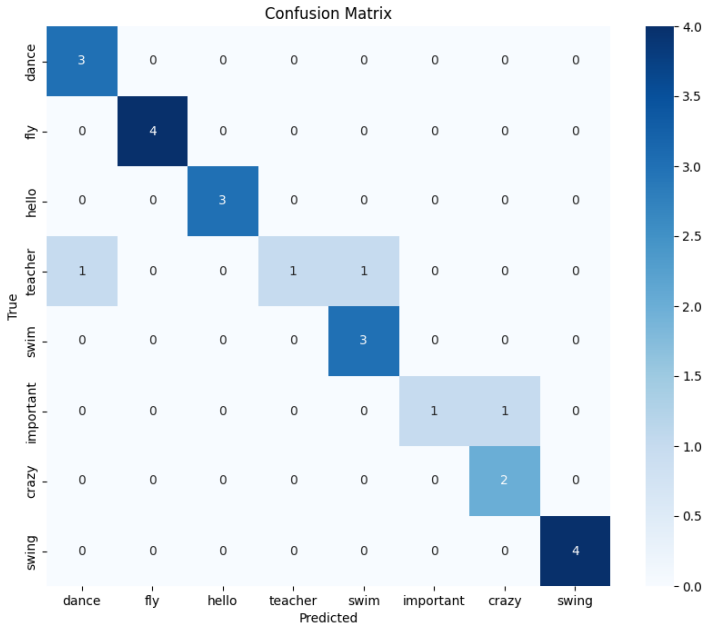

Figure 72: The confusion matrix for the fourth experiment's Slow Fast Neural Network, using all labels and the BODY AND HANDS data processing technique.

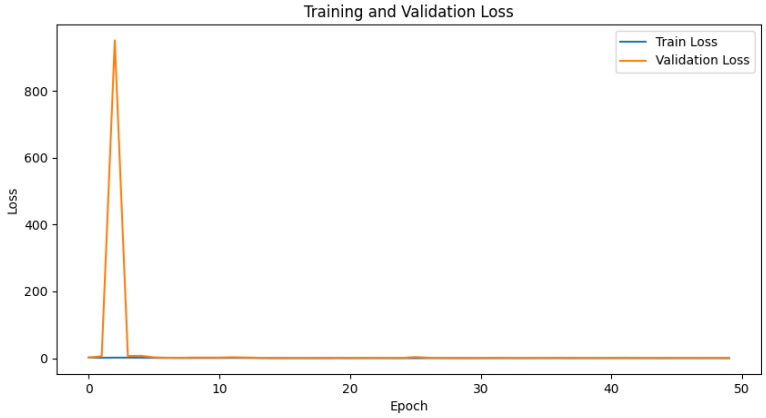

Figure 73: The loss function graph obtained from training the Slow Fast Neural Network in the fourth experiment using all labels and the BODY AND HANDS data processing technique.

C.4.2 3DCNN using the ALL data processing

Figure 74 portrays the confusion matrix for the 3DCNN of the fourth experiment trained with the body gestures labels of Table 4 and applying to the input data the ALL data processing technique explained on Section 3.2.2. Additionally, Figure 75 depicts the model's loss function during training. This particular model achieved an accuracy level of 76.64%.

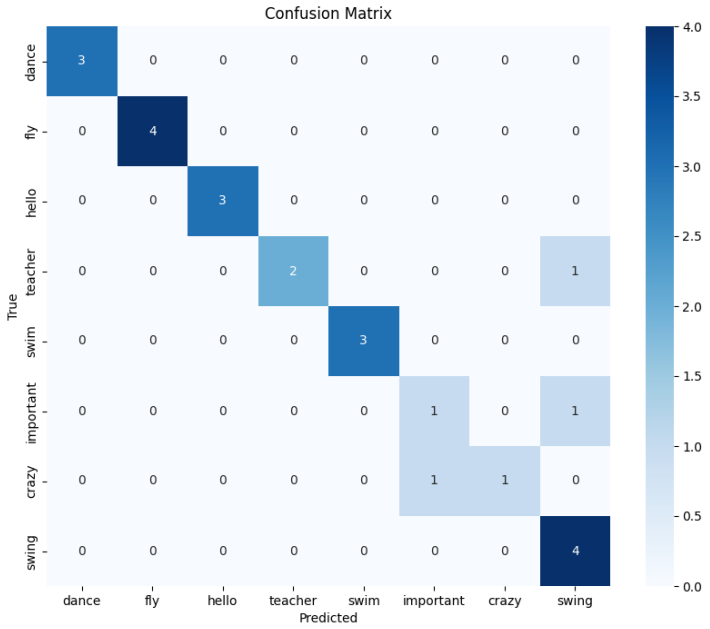

Figure 74: The confusion matrix for the fourth experiment's 3DCNN, using all labels and the ALL data processing technique.

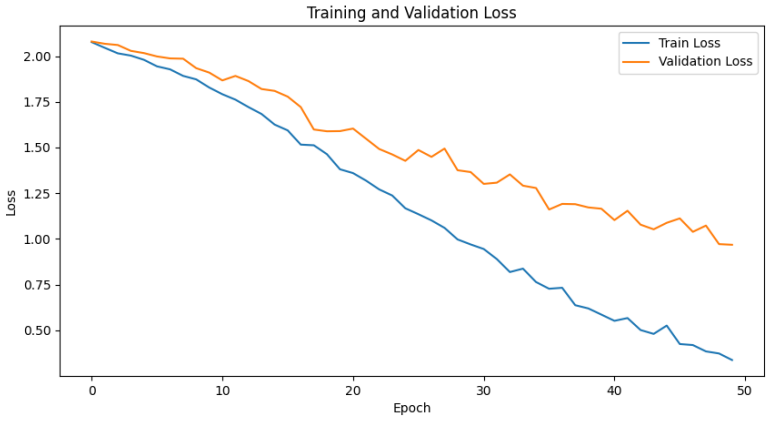

Figure 75: The loss function graph obtained from training the 3DCNN in the fourth experiment using all labels and the ALL data processing technique.

D Fifth experiment

D.1 SlowFast neural network model

Figure 76 shows the confusion matrix for the 3DCNN of the fifth experiment trained with 50 labels applying to the input data the ALL data processing technique explained on Section 3.2.2. Additionally, Figure 77 depicts the model's loss function during training. This particular model achieved an accuracy level of 62.43%.

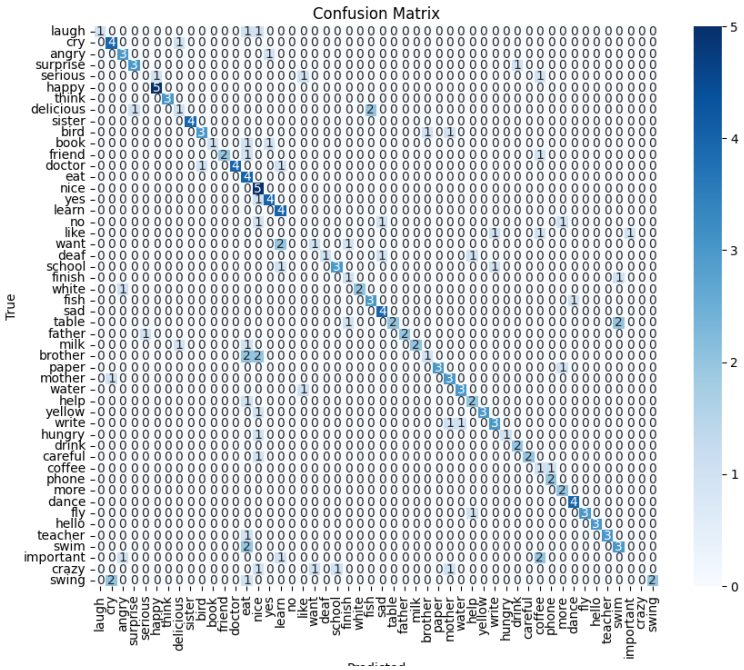

Figure 76: The confusion matrix for the fifth experiment's Slow Fast Neural Network, using all labels and the ALL data processing technique.

Figure 77: The loss function graph obtained from training the Slow Fast Neural Network in the fourth experiment using all labels and the ALL data processing technique.

D.2 3DCNN Model

Figure 78 portrays the confusion matrix for the 3DCNN of the fifth experiment trained with 50 labels and applying to the input data the ALL data processing technique explained on Section 3.2.2. Additionally, Figure 79 depicts the model's loss function during training. This particular model achieved an accuracy level of 67.72%.

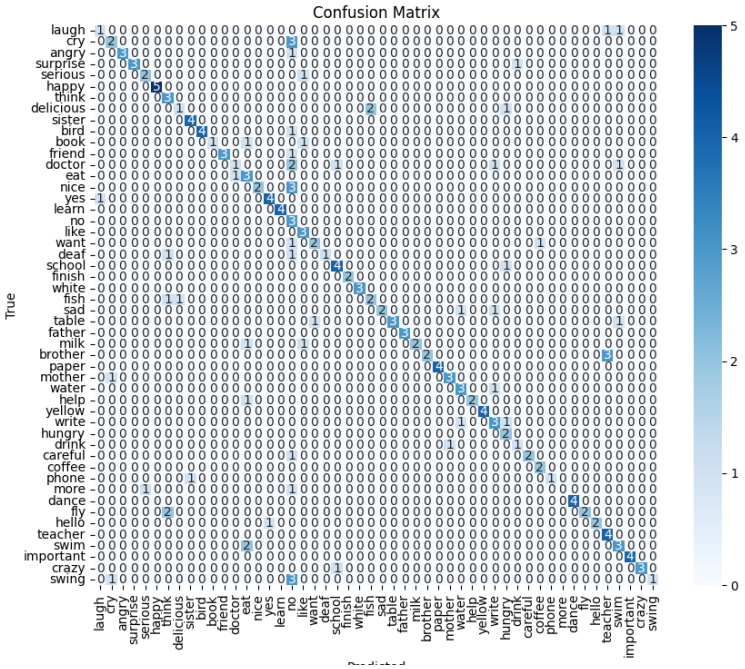

Figure 78: The confusion matrix for the fifth experiment's 3DCNN, using all labels and the ALL data processing technique.

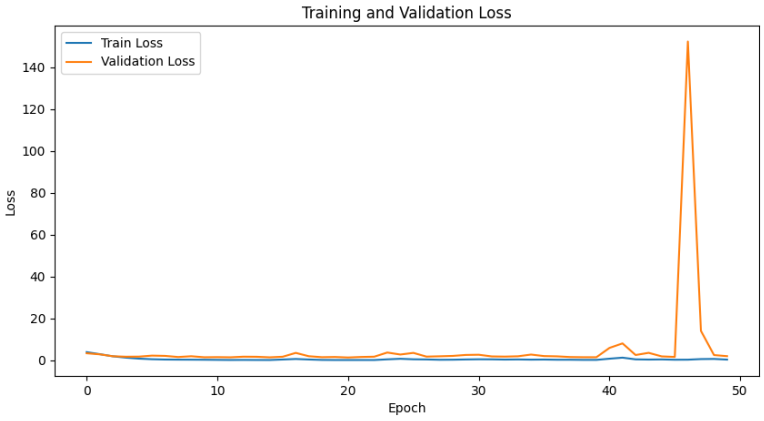

Figure 79: The loss function graph obtained from training the 3DCNN in the fifth experiment using all labels and the ALL data processing technique.

References

[1] Ankita Wadhawan and Parteek Kumar. Sign language recognition systems: A decade systematic literature review. *Archives of Computational Methods in Engineering*, 28:785 – 813, 2019.

[2] Alexey Karpov, Irina Kipyatkova, and Milos Zelezny. Automatic technologies for processing spoken sign languages. *Procedia Computer Science*, 81:201–207, 2016. SLTU-2016 5th Workshop on Spoken Language Technologies for Under-resourced languages 09-12 May 2016 Yogyakarta, Indonesia.

[3] Barbara R Schirmer. *Psychological, social, and educational dimensions of deafness.* Allyn & Bacon, 2001.

[4] Annalene Van Staden, Gerhard Badenhorst, and Elaine Ridge. The benefits of sign language for deaf learners with language challenges. *Per Linguam: a Journal of Language Learning= Per Linguam: Tydskrif vir Taalaanleer*, 25(1):44–60, 2009.

[5] Cemil Oz and Ming C. Leu. Linguistic properties based on american sign language isolated word recognition with artificial neural networks using a sensory glove and motion tracker. *Neurocomputing*, 70(16):2891–2901, 2007. Neural Network Applications in Electrical Engineering Selected papers from the 3rd International Work-Conference on Artificial Neural Networks (IWANN 2005).

[6] Chao Sun, Tianzhu Zhang, Bingkun Bao, and Changsheng Xu. Latent support vector machine for sign language recognition with kinect. *2013 IEEE International Conference on Image Processing*, pages 4190–4194, 2013.

[7] Chao Sun, Tianzhu Zhang, Bing-Kun Bao, Changsheng Xu, and Tao Mei. Discriminative exemplar coding for sign language recognition with kinect. *IEEE Transactions on Cybernetics*, 43(5):1418–1428, 2013.

[8] Pat Jangyodsuk, Christopher Conly, and Vassilis Athitsos. Sign language recognition using dynamic time warping and hand shape distance based on histogram of oriented gradient features. In *Proceedings of the 7th International Conference on PErvasive Technologies Related to Assistive Environments*, PETRA '14, New York, NY, USA, 2014. Association for Computing Machinery.

[9] Jian Wu, Zhongjun Tian, Lu Sun, Leonardo Estevez, and Roozbeh Jafari. Real-time american sign language recognition using wrist-worn motion and surface emg sensors. In *2015 IEEE 12th International Conference on Wearable and Implantable Body Sensor Networks (BSN)*, pages 1–6, 2015.

[10] Panupon Usachokcharoen, Yoshikazu Washizawa, and Kitsuchart Pasupa. Sign language recognition with microsoft kinect's depth and colour sensors. *2015 IEEE International Conference on Signal and Image Processing Applications (ICSIPA)*, pages 186–190, 2015.

[11] Celal Savur and Ferat Sahin. Real-time american sign language recognition system using surface emg signal. *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*, pages 497–502, 2015.

[12] Chao Sun, Tianzhu Zhang, and Changsheng Xu. Latent support vector machine modeling for sign language recognition with kinect. *ACM Trans. Intell. Syst. Technol.*, 6(2), mar 2015.

[13] Anup Kumar, Karun Thankachan, and Mevin M. Dominic. Sign language recognition. *2016 3rd International Conference on Recent Advances in Information Technology (RAIT)*, pages 422–428, 2016.

[14] D. Anil Kumar, Polurie Venkata Vijay Kishore, A. S. Chandrasekhara Sastry, and P. Reddy Gurunatha Swamy. Selfie continuous sign language recognition using neural network. *2016 IEEE Annual India Conference (INDICON)*, pages 1–6, 2016.

[15] Celal Savur and Ferat Sahin. American sign language recognition system by using surface emg signal. In *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 002872–002877, 2016.

[16] Duaa AlQattan and Francisco Sepulveda. Towards sign language recognition using eeg-based motor imagery brain computer interface. *2017 5th International Winter Conference on Brain-Computer Interface (BCI)*, pages 5–8, 2017.

[17] M.K. Bhuyan, D. Ghosh, and P.K. Bora. Feature extraction from 2d gesture trajectory in dynamic hand gesture recognition. In *2006 IEEE Conference on Cybernetics and Intelligent Systems*, pages 1–6, 2006.

[18] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE Computer Society Conference on Computer*

Vision and Pattern Recognition (CVPR'05), volume 1, pages 886–893 vol. 1, 2005.

[19] Mark Borg and Kenneth P. Camilleri. Phonologically-meaningful subunits for deep learning-based sign language recognition. In Adrien Bartoli and Andrea Fusiello, editors, *Computer Vision – ECCV 2020 Workshops*, pages 199–217, Cham, 2020. Springer International Publishing.

[20] Oscar Koller, Jens Forster, and Hermann Ney. Continuous sign language recognition: Towards large vocabulary statistical recognition systems handling multiple signers. *Computer Vision and Image Understanding*, 141:108–125, 2015.

[21] Jiangbin Zheng, Zheng Zhao, Min Chen, Jing Chen, Chong Wu, Yidong Chen, Xiaodong Shi, and Yiqi Tong. An improved sign language translation model with explainable adaptations for processing long sign sentences. In *Computational Intelligence and Neuroscience*, volume 2020, page 11, 2020.

[22] Muneer Al-Hammadi, Ghulam Muhammad, Wadood Abdul, Mansour Alsulaiman, Mohamed A. Bencherif, and Mohamed Amine Mekhtiche. Hand gesture recognition for sign language using 3dcnn. *IEEE Access*, 8:79491–79509, 2020.

[23] Ahmed Hassan, Ahmed Elgabry, and Elsayed Hemayed. Enhanced dynamic sign language recognition using slowfast networks. In *2021 17th International Computer Engineering Conference (ICENCO)*, pages 124–128. IEEE, 2021.

[24] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Antonino Furnari, Jian Ma, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, and Michael Wray. Rescaling egocentric vision. *CoRR*, abs/2006.13256, 2020.

[25] Dongxu Li, Cristian Rodriguez, Xin Yu, and Hongdong Li. Word-level deep sign language recognition from video: A new large-scale dataset and methods comparison. In *The IEEE Winter Conference on Applications of Computer Vision*, pages 1459–1469, 2020.

[26] Hamid Vaezi Joze and Oscar Koller. Ms-asl: A large-scale data set and benchmark for understanding american sign language. In *The British Machine Vision Conference (BMVC)*, September 2019.

[27] Vassilis Athitsos, Carol Neidle, Stan Sclaroff, Joan Nash, Alexandra Stefan, Quan Yuan, and Ashwin Thangali. The american sign language lexicon video dataset. In *2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 1–8. IEEE, 2008.

[28] Haoqi Fan, Yanghao Li, Bo Xiong, Wan-Yen Lo, and Christoph Feichtenhofer. Pyslowfast. `https://github.com/facebookresearch/slowfast`, 2020.

[29] Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, et al. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017.

[30] Joao Carreira, Eric Noland, Andras Banki-Horvath, Chloe Hillier, and Andrew Zisserman. A short note about kinetics-600. *arXiv preprint arXiv:1808.01340*, 2018.

[31] Gunnar A Sigurdsson, Gül Varol, Xiaolong Wang, Ali Farhadi, Ivan Laptev, and Abhinav Gupta. Hollywood in homes: Crowdsourcing data collection for activity understanding. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*, pages 510–526. Springer, 2016.

[32] Chunhui Gu, Chen Sun, David A Ross, Carl Vondrick, Caroline Pantofaru, Yeqing Li, Sudheendra Vijayanarasimhan, George Toderici, Susanna Ricco, Rahul Sukthankar, et al. Ava: A video dataset of spatio-temporally localized atomic visual actions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6047–6056, 2018.

[33] TorchVision maintainers and contributors. Torchvision: Pytorch's computer vision library. `https://github.com/pytorch/vision`, 2016.

[34] Du Tran, Heng Wang, Lorenzo Torresani, Jamie Ray, Yann LeCun, and Manohar Paluri. A closer look at spatiotemporal convolutions for action recognition. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 6450–6459, 2018.

[35] Haoqi Fan, Tullie Murrell, Heng Wang, Kalyan Vasudev Alwala, Yanghao Li, Yilei Li, Bo Xiong, Nikhila Ravi, Meng Li, Haichuan Yang, Jitendra Malik, Ross Girshick, Matt Feiszli, Aaron Adcock, Wan-Yen

Lo, and Christoph Feichtenhofer. PyTorchVideo: A deep learning library for video understanding. In *Proceedings of the 29th ACM International Conference on Multimedia*, 2021. `https://pytorchvideo.org/`.