KLE Society's
KLE Technological University, Hubballi.

A Minor Project Report

on

Indian Sign Language Recognition and Translation

submitted in partial fulfillment of the requirement for the degree of

Bachelor of Engineering

in

Computer Science and Engineering

Submitted by

Akshata Badni	01FE20BCS005
Ranjeeta Angadi	01FE20BCS027
Neha Jain	01FE20BCS083
Madhura Patil	01FE20BCS187

Under the guidance of
Dr. Sujata C

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

Huballi – 580031

2022-2023

KLE Society's
KLE Technological University, Hubballi.

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that Minor Project titled Indian Sign Language Recognition and Translation is a bonafied work carried out by the student team comprising of Akshata Badni (01FE20BCS005), Ranjeeta Angadi (01FE20BCS027), Neha Jain (01FE20BCS083), Madhura Patil (01FE20BCS187) for partial fulfillment of completion of sixth semester B.E. in Computer Science and Engineering during the academic year 2022-2023.

Guide — Dr. Sujata C

Head, SoCSE — Dr. Meena S. M

Viva -Voce:

Name of the Examiners	Signature with date
1.	
2.	

Acknowledgement

We would like to thank our faculty and management for their professional guidance towards the completion of the project work. We take this opportunity to thank Dr. Ashok Shettar, Vice-Chancellor, Dr. B.S Anami, Registrar, and Dr. P.G Tewari, Dean Academics, KLE Technological University, Hubballi, for their vision and support.

We also take this opportunity to thank Dr. Meena S. M, Professor and Head, SoCSE for having provided us direction and facilitated for enhancement of skills and academic growth.

We thank our guide Dr. Sujata C, SoCSE for the constant guidance during interaction and reviews.

We extend our acknowledgement to the reviewers for critical suggestions and inputs. We also thank Project Co-ordinator Mr. Uday N.Kulkarni and Mr. Guruprasad Konnuramath for their support during the course of completion.

We express our gratitude to the Prinicipal and teachers of Priyadarshini Deaf And Dumb Residential School Gurunath Nagar, Anand Nagar Hubballi, for co-operating and helping us in creating the dataset.

We express gratitude to our beloved parents for constant encouragement and support.

Neha Jain - 01FE20BCS083
Akshata Badni - 01FE20BCS005
Madhura Patil - 01FE20BCS187
Ranjeeta Andagi - 01FE20BCS027

ABSTRACT

Sign Language is the medium of communication in the Deaf and Hard of Hearing community (DHH community). It uses hand gestures and facial expressions as the main tools for the communication between people. Sign Language is as independent as any other language. It has its own linguistics and grammar that is different from spoken language. Just like any other language it takes mush time and effort to learn hence discouraging the larger population from learning the language. It becomes difficult for communication between DHH community and others. This makes them isolated in the society.

Sign gestures can be classified as static and dynamic. Static sign recognition is comparatively simpler than dynamic gestures but both are crucial tothe human community. Hence, we can use deep learning techniques to recognize the gestures and after the model recognizes the gesture the corresponding text is generated. This automation of sign videos to text will bridge the gap between the DHH community and others.

Currently there are limited publicly available datasets on Indian Sign Language for recognition tasks. In this work we present the Kannada Sign Language Dataset-KSL, a dataset at Character, Word and Sentence for the kannada regional language. This dataset contains across 925 videos, with the total number of 113 classes. KSL is recorded with the help of experienced signers from a deaf and dumb school. We compare our dataset with existing INCLUDE-50 ISL dataset. The dataset is trained and evaluated on an attention based BiLSTM model combined with different augmentation and feature extraction methods. The model achieves an accuracy of 81.7 percent on this dataset. We also extend our experiment on INCLUDE-50 dataset with this model and report an improvement in the performance from 73.9 percent to 93.4 percent.

Keywords : *Sign Language Recognition, Sign Language Translation, Indian language Dataset, INCLUDE, Kannada Sign Language, Augmentation, MobileNet, BiLSTM,*

CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

INTRODUCTION

Languages are a fundamental requirement for communication and the exchange of ideas, feelings, and knowledge.People with hearing or speech impairments utilize a variety of gestures to communicate with others because language is spoken.As a result, it is called 'Sign Language'. It has its particular patterns and variables. There are about 300 distinct sign languages used across the world such as Chinese Sign Language (CSL), French Sign Language (FSL), Japanese Sign Language (JSL), British, Australian, and New Zealand Sign Language (BANZSL), Indian Sign Language (ISL) American Sign Language (ASL) etc. The gestures of signs change from country to country. For example some countries use single hand for making gestures while some countries make use of both the hand.

The basic structure of sign language is 'time + topic + comment + place'. Since sign language is visual based language, facial expressions and body language matters a lot. Our facial expressions frame how our sentences are stated. Body language includes hand movements, pose and upper body movements. Learning sign language is not similar to learning spoken languages. It requires more time and effort to master the visual gestures. Hence it will always be difficult for someone who uses sign language to communicate with people who do not understand the gestures.

Sign-to-text translation is a machine translation architecture that will translate the given sign language image of some gesture or a video clip into the text of the target language in real-time. This will help the deaf and speech-impaired community to connect with society and interact with other people. It will also help those who wish to learn sign language. The model can eliminate the middleman, who generally acts as a medium of translation hence bridging the gap between hearing impaired people and others.

1.1 Motivation

A total of 430 million people, or more than 5 Percent of the world's population, need rehabilitation for hearing loss, and roughly 63 million of those come from India. The communication barrier contributes to a very high unemployment rate in the deaf and hard of hearing (DHH) communities. As a result of their failure to develop foundational skills at a young age, members of the DHH community find it extremely challenging to pursue an education.

Sign language corpora are limited available, and the few that are often have little or no annotation. Annotation is required because the corpora are typically in the form of sign language videos due to the lack of a standardized written form for SLs.

Language translation using computer vision techniques has been a growing topic in research nowadays. Like any other natural language, sign language also has its own grammar and linguistic properties, which are very different from all other languages. This makes the translation process complex. It cannot be translated into a spoken language word-by-word.

Every gesture in sign language will have a meaning and it will translate to a single word or a phrase in spoken language. In the case of sign language videos, such one-to-one translation will not provide meaningful output. Hence, understanding the grammar of both the source and target language will become necessary.

The study of language translation using computer vision methods is becoming increasingly popular.Sign language has its own syntax and linguistic characteristics that set it apart from all other languages, just like any other natural language.The translation procedure is complicated as a result.It cannot be verbatim translated into a spoken language.Every sign language motion has a specific meaning that corresponds to a particular word or phrase in spoken language.Such one-to-one translation will not produce relevant results for sign language videos.Consequently, it will be important to comprehend both the source language's and the target language's grammar.

1.2 Literature Survey

This particular part of the survey throws light on existing models in the field of Sign language recognition and translation.The proposed work by all the authors helped us alot to get sucseed in our project.

Kagalkar. R. M [1] author mainly utilizes two algorithms Namely HOG: Feature Extraction[2] and SVM: [3] Classifiction algorithms for training and testing. Starting with the traning part, they extract the frames, perform preprocessing to remove the noice and bluriness effect using media filter,detect edge(Canny edge detection),extract the features with HOG technique.The features extracted are stored into database in CSV file along with the associated class labels and also marked features.Next the testing part, here the model will take the data from databases and apply the SVM classification algorithm and produce the appropriate output for the given input video.

Poonia, R. C. [4] discussed about the drawbacks and proposed a solution for the problems. Firstly a sequential translated model was tested on CNN-RNN,it extracted spacial and temporal features from the given input video. The model took long time to train and the performance was not efficient. Secondly Pose-Based feature extractor was used to reduce the limitation of CNN-RNN. Pose estimation tracked the signers hand movements and facial expressions. The extracted features are fed into LSTM classifier [5]. The model captured only the major hand movements and failed to capture the different signer speed, that is why regarding this failure, the inception was chosen to extract around 2048 features from the video hence making it more efficient compared to VGG16-19.THe extracted features are then converted into 2D vector known as the vector map and fed into LSTM classifier, which will learn the features and translate them into corresponding text.

Alawwad, R. A. [6] The author has built a Dataset with non-uniform background and no color restrictions.Images with different hand sizes and skin tones50 identical Arabic signs Key property- Semantic meaning of the gesture.The author has proposed a methodology using Faster-RCNN.In this work they investigate two things,first it will associate the deep VGG-16 to the faster RCNN, and next it is relied on Resnet. An image is fed as an input to a pre trained CNN(VGG-16 or ResNet) to generate feature maps. These are then fed to Region Proposal Network to generate potential hand gestures.The proposals generated by RPN are conveyed to the ROI pooling layer alongside the feature map generated by CNN.The scaled feature maps then are fed to Fully connected layer for classification.

Yuan.T. [7] Openpose method is used to extract Body skeletal and facial representation.The features representations are used as an input features to the recurrent model. The S2VT(seq2seq v2t) framework first encodes the frames, one frame at a time to the first layer of a two-layer LSTM, where f is of variable length. This latent representation is then decoded into natural language sentence one word at a time, feeding the output of one time step into

the second layer of the LSTM in the subsequent time step. Results show that the current deep learning methods are able to over fit training splits but is unable to generalize well on more difficult CSLD dataset.

Kagalkar.RM [8] The author propose an approach that defines new representative feature sign curvilinear coding, which extracts the major features from sample, uses leap forward tracing algorithm and region close area feature and process for character generation. From this survey we can infer that there is no publicly dataset available and built on local language especially in kannada. We can also infer that the Inception and MobileNet are the main modern deep neural networks which extract the maximum features from all the frames and also the LSTM which outshines at classification techniques.

1.3 Datasets on Sign Lnaguage

- Several datasets for sign languages around the world have been built.The INCLUDE dataset [9] that includes 4,287 videos and 0.27 million frames, with 263 word signs from 15 different word groups. To offer a close match to the environment, INCLUDE is recorded with the assistance of skilled signers. To define INCLUDE-50, a subset of 50 word signs from various word categories were selected.

- Another dataset RWTH-Boston-50 [10] [11] is another 50 class ASL lexicon dataset that is fairly popular.

- RWTH-PHOENIX-Weather 2014 [12] [13] is a German sign language dataset aimed at continuous sign language recognition. The dataset has over a million frames and a vocabulary size of 1,081 unique words. The dataset is recorded from a public television weather broadcast and has sentences performed by 9 unique signers.

- Ko etal. [14] propose a dataset for continuous Korean Sign Language recognition called KETI, [15]with over 14,000 videos and 100 sentences.

- Nandy et al.'s [16] [17] dataset comprises a set of about 600 vedios across 22 classes. These videos were trimmed to visible only the hands and are in grayscale.

- MS-ASL is another dataset for ASL Lexicon Recognition [18] [19] Over 222 signers contributed over 1,000 classes of signs to the MS-ASL collection. In comparison to the ASLLVD dataset, this offers more diversity.

- Kishore et al.[20]proposes a dataset that has a considerable vocabulary of videos (800 videos across 80 classes) but have constraints imposed on how the videos are recorded.

Thus, research done in Indian Sign Language especially in Regional Kannada Language is less.In India, there is distinctive gesture based communication watched for each region locale. It is henceforth troublesome for one area individual to convey to other utilizing a signature image. There is a need for a unified Sign Language which will be helpful for most of the DHH community. Therefore to overcome this troublesome, we have built a KSL dataset at character, word and sentence in Regional kannada language.

1.4 The KSL Dataset

The main contribution of this project is to build the dataset in Indian Sign Language especially in Regional Kannada Language. The videos were recorded with the help of experienced signers for hearing impaired people .The dataset was created by recording at a school for the deaf with the help of senior Lectures. This was done by following all administrative rules, procedures and with the official support of the school. All the lectures were experienced signers in ISL having completed several years of experience in this field of education.

Figure 1.1: Frames of KSL Dataset

1.5 Problem Statement

To address the problem of Indian Sign Language Recognition at character, word, and sentence level for a given sequence of signs.

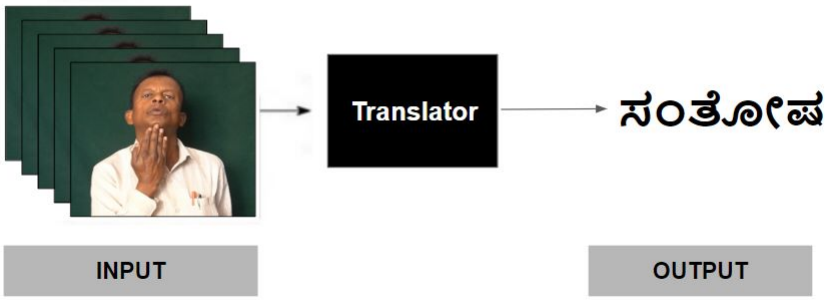

Figure 1.2: Kannada Sign Language recognition

1.6 Applications

- Education: Sign language recognition technology can be used to create educational tools for deaf and hard of hearing students, such as interactive learning games and tutorials.
- Translation: Sign language recognition can be used to automatically translate sign language into spoken language or text, making it easier for deaf individuals to communicate with people who do not know sign language.
- Accessibility: Sign language recognition can be used to provide real-time captioning of spoken content in live events, such as lectures or speeches, making them accessible to deaf or hard of hearing individuals.

1.7 Objectives and Scope of the project

A project involving the recognition and translation of sign language in Kannada, may have a very broad scope and may entail working with specialists in computer science, linguistics, and sign language interpretation. A system that can improve communication between the deaf and hard-of-hearing people and the larger Karnataka, India, society would be the aim of such a project.

1.7.1 Objectives

- To build a dataset in Indian Sign Language(ISL) at Character, Word and Sentence for the kannada regional language.
- To build a model for translation of given ISL signs to kannada text.
- Compare the proposed work with state-of-the-art.

1.7.2 Scope of the project

In this project, we have implemented a sign language recognition and translation system using our KSL dataset.
The dataset is not just limited to word level but also to character and sentence level, providing a broader scope of communication with the hearing impaired and thus bridging the communication gap with a unified dataset.

Possible extensions to this work would be collecting more data and building a gesture recogniton system on an adequate vocabulary of ISL. Further we also plan to build a bidirectional sign language translating system i.e a system converts a given sign to its corresponding text and vice-versa.

1.7.3 Dataset Building

- Videos were captured using Canon EOS 80D 24.2MP Digital SLR Camera.
- Dataset is built in 3 different Levels: Character,word and sentence Level
- A Total of 868 videos were made.
- Total number of classes - 113 classes

Figure 1.3: Visited Priyadarshini deaf and dumb school

Chapter 2

REQUIREMENT ANALYSIS

A specification of software requirements is a document that outlines the functions and performance standards for the software.It also outlines the functionality the product needs to have in order to meet the needs of both business and user stakeholders.

2.1 Functional Requirements

Functional requirements are a critical aspect of developing a sign language recognition and translation system. These requirements define the features and capabilities that the system must have to meet the needs of its users.Overall, the functional requirements of a sign language recognition and translation system are designed to ensure that it meets the needs of its users, is effective in facilitating communication, and is easy to use and access for all

- The system shall preprocess as per the requirements of the model.
- The model must extract and learn the features from input data.
- Given a new video the model shall be able to predict the output.

2.2 Non Functional Requirements

Non-functional requirements are an essential part of the design and development of a sign language recognition and translation system. These requirements describe the qualities and characteristics that the system must possess to operate effectively and efficiently. Non-functional requirements ensure that the system is not only effective in its core functions but also operates efficiently, securely, and reliably to meet the needs of its users.

- Model accuracy should be >=71(Baseline Accuracy).
- The model must require GPU and other computation requirements to work.
- The model should be trained within 2 or 3 hours.
- The model should produce the output in kannada label.

2.3 Hardware Requirements

- Processor - Intel Core i7-6700 CPU @ 3.40GHz x 8.
- GPU - NV167, usage = 2GB.

2.4 Software Requirements

- OS - Ubuntu 20.04.5 LTS
- Tensorflow 2.10, Pytorch 1.4

Chapter 3

SYSTEM DESIGN

Translating Sign language videos to text is sequential task. Our objective is to improve an existed infamous BiLSTM model by making changes to dataset before obtaining the embeddings. Before directly extracting the features from the frames, we made certain modification so that it leads to extraction of important features thus improving the model's overall learning. Moreover, these methods are suitable for both big and small datasets and can also used as general preprocess steps before obtaining the embeddings for the train networks.

The different combination of below mentioned methods were used for our dataset:

- Obtaining keypoints of video frames. The keypoints are the coordinate points of few fixed body parts of the subject in each and every frame. Bodyparts like hand, arm, face, upperbody, pose etc are considered. The advantage of this method is that it can be more efficient in terms of computational resources, as the CNN is only being applied to specific regions of interest (ROIs) within each frame, rather than the entire frame. This can be especially useful when processing large amounts of video data.
- Data Augmentation methods like rotation, gauss sampling, cutout, downsampling and upsampling were used to build volume of the dataset. These augmentations were performed on the keypoints obtained from previous method.
- The augmented keypoints are given to a CNN network to extract features. Lastly SMOTE augmentation method was applied to this CNN features. The purpose was to obtain a balanced data features. The performance of the model enhanced to great extent after applying the augmentation.

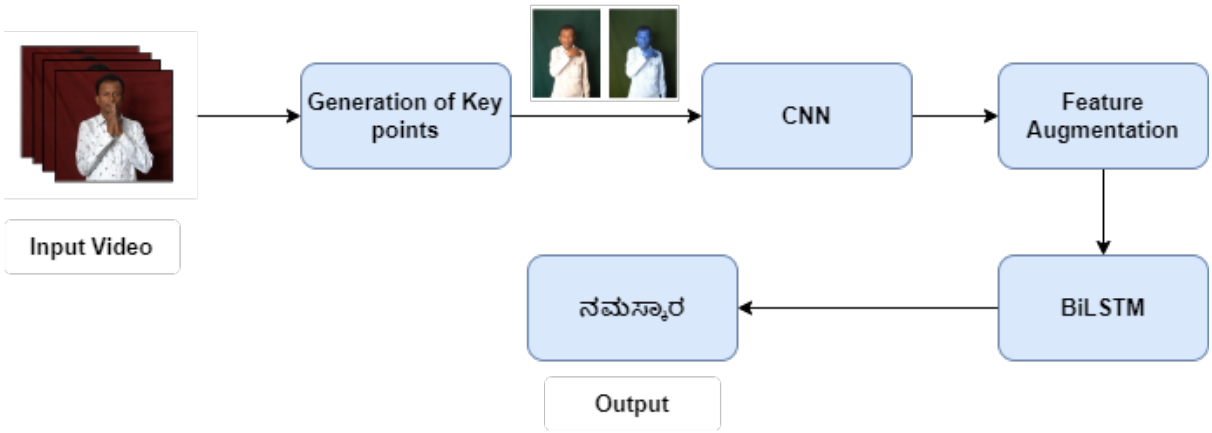

Figure 3.1: System Design

3.1 Architecture Design

Every frame passes through Media Pipe, which will produce the keypoints. For each frames, it will generate 468 facial landmarks, 21 hand landmarks on each hand, and 33 upper body landmarks. To extract features, a channel-wise concatenation is performed and put into the MobileNetV2 model. A BiLSTM receives the extracted features as input. For classification, the hidden states from LSTM cells are flattened and passed through two layers: a fully connected layer and a softmax layer.

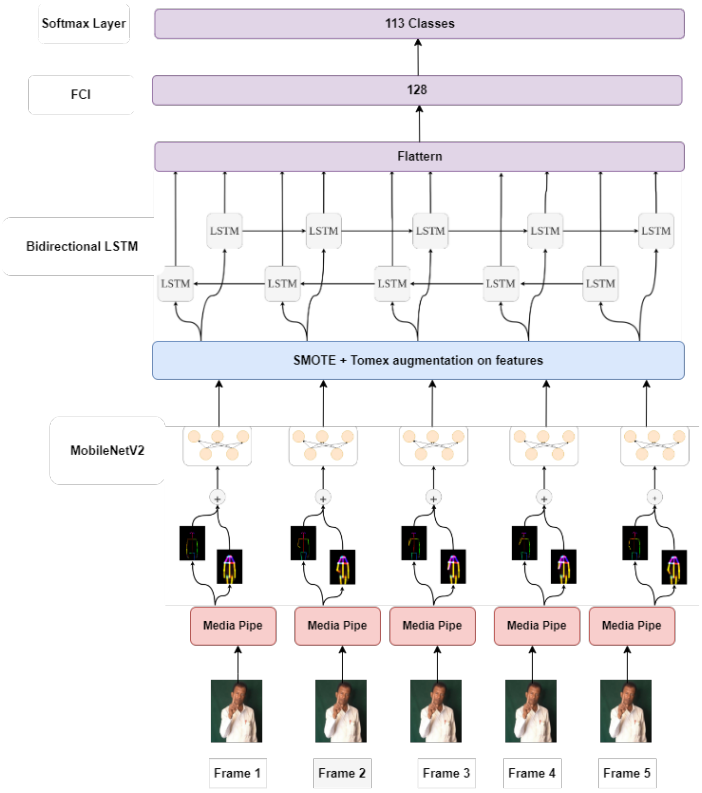

Figure 3.2: Overview of Architecture Design

3.1.1 Preprocessing

preprocess aims to improve the frames which will be feeded as the input to the model,In our work the frames are resized from 720 x 1280 to 1080 x 1920 using lanczos interpolation method, frames were Rotated by 90 degrees(from landscape to portrait).and also Reduced fps from 60 to 30 ,for the good performance of the model we have reduced the noice by muting the vedios. we Split the videos into clips of a particular class.Finally the dataset is further splitted into training validation and testing datasets.

3.1.2 Labeling

- Labels: The model should be trained to recognise the videos for kannada labels. The labels will be generated manually for every video. Each label will be used as a key and will have an integer value.

- Encoding: The labels generated will be encoded using encoding utf-8 or NLTK toolkit. The input to the model will be in the form of integers and output will be vector of probabilities for each sign.

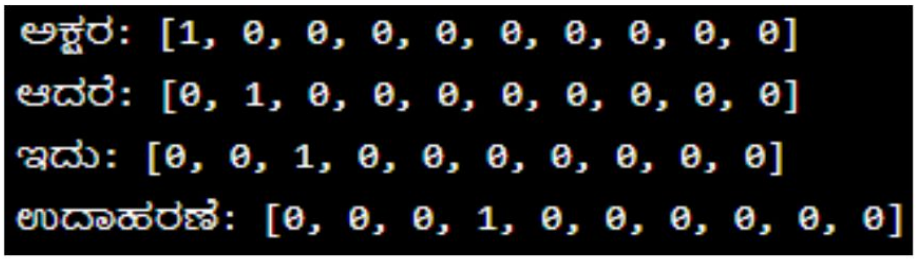

Figure 3.3: Labeling

3.1.3 Generating the keypoints

This is a crucial step in many machine learning applications,it is a component that takes input data.The Light Intensity,Hand-hand overlap and Hand-face overlap are the parameters were tested when choosing a suitable method for extracting key points.Mediapipe is a popular open-source framework developed by Google that offers various pre-trained models for computer vision tasks such as object detection, hand tracking, and pose estimation.
For pose estimation, Mediapipe provides a pre-trained model that detects and tracks multiple human body keypoints in real-time. The model is based on the EfficientDet-D0 architecture and uses a top-down approach to predict 21 keypoints for each hand and 33 keypoints for each detected person, for the pose. this shows the implementation details for generating keypoints for hands using Mediapipe hand tracking(OpenCV). Same algorithm is used to generate keypoints for the pose. Key points are selected for randomly selected frames, and these keypoints of the body and pose are marked by blue-points.

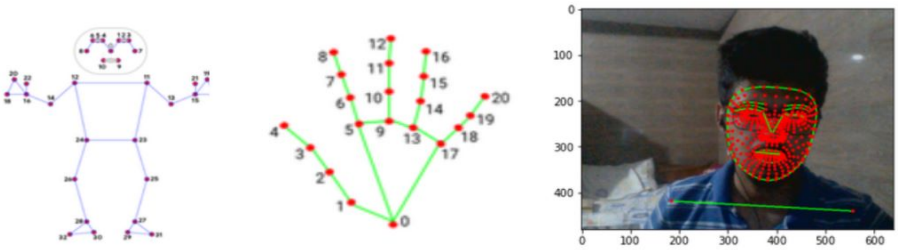

Figure 3.4: Generation of keypoints

3.1.4 Extracting CNN features

Feature extraction is a crucial step in many machine learning applications,it is a component that takes input data and transforms it into a set of features`MobileNetV2_100` is a convolutional neural network architecture that was designed for efficient image classification on mobile and embedded devices. `MobileNetV2_100` is a variant of the MobileNetV2 architecture, which was introduced in the same paper. The "100" in the model's name refers to the width multiplier used in the architecture. Specifically, the width multiplier is set to 1.0 in the `MobileNetV2_100` model, which means that the number of filters in each layer is the same as in the original MobileNetV2 architecture.

3.1.5 Feature Augmentation

Data augmentation is a set of technique that involve generating new data points from existing data to enhance the amount of data.Our objective is to improve an existed infamous BiLSTM model by making changes to dataset before obtaining the embeddings. Before directly extracting the features from the frames, we made certain modification so that it leads to extraction of important features thus improving the model's overall learning.We perform the following augmentations on the input videos:

- Combination of oversampling and undersampling methods is applied to overcome the imbalanced dataset problem.
- Smote, an oversampling technique is used to generate synthetic data for the minority class based on the distance of each data being selected and the k nearest neighbours.
- Tomek Links is then used to remove the data from the majority class that is closest with the minority class data.

Thus,a single video has been undertaken through these augmentations to generate several variants.Augmenting the dataset leads to good performance and improve accuracy of the models on the KSL dataset.

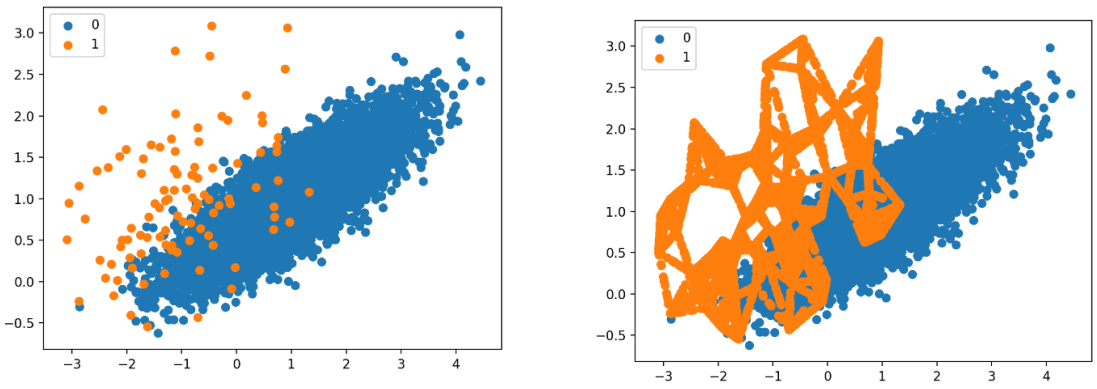

Figure 3.5: Visualisation of SMOTE augmentation for a binary data

The figure shows the imbalanced binary data. SMOTE methods identifies the minority class and make a cluster of both classes. The method uses mean or median between every two points in the minority class and generates a new point thus increasing the total number of points and giving a balanced data. Similarly, for images and videos, the feature points of the minority class can be augmented to obtain a balanced data.

3.2 Dataset Description

There is a limited ISL dataset available publicly to perform sign language recognition using deep learning techniques. This data is not sufficient to obtain an effective model that can be used in real time. But in recent years, with advancement in the field of AI, the importance of abundant data for training models has grown. Our contribution is to build an ISL dataset in regional language and further extend the number of classes and videos to make it a sufficient dataset for training any sign recognition models.

The dataset consists of video clips recorded by the expert signers for regional kannada language. It consists of video clips of average length of 4 seconds for every class. The classes are categorised into characters, words and sentences based on the complexity of the model. Every class depicts a character, word or a sentence in kannada script. The model has to translate the video directly to the text in kannada and not having english words as intermediate text.

- In character level, every alphabet and number in kannada are taken as individual classes.
- In word level, few examples in specific subjects have been taken as separate classes.
- In sentence level, few statements containing more than two words that are practised in daily use are considered as individual class.
- A total of 868 videos were built covering 113 classes.

	Character Level	**Word Level**	**Sentence Level**
Classes	61 letters	36	16
Number of Videos	8 for each class	7 for each class	8 for each sentence
Description	1. ಕನ್ನಡ ಸ್ವರಗಳು (Kannada Swaragalu) 2. ಕನ್ನಡ ವ್ಯಂಜನಗಳು (Kannada Vyanjanagalu) 3. ಕನ್ನಡ ಅಂಕಿಗಳು (Kannada sanke galu)	1.ಭಾವನೆಗಳು(emotions) 2.ದಿನಾಂಕ(date and time) 3. ಋತುಗಳು(seasons) 4. ಬಣ್ಣ(colours) 5. ಸರ್ವನಾಮ(Pronouns)	1. ನನಗೆ ಕನ್ನಡ ಎಂದರೆ ಇಷ್ಟ. 2. ನೀವು ನನಗೆ ಸಹಾಯ ಮಾಡುವಿರಾ?

Figure 3.6: Dataset description

Chapter 4

IMPLEMENTATION

This chapter gives a brief description about implementation details of the system by describing each component with its code skeleton in terms of algorithm.

4.1 Generating keypoints

Mediapipe is a popular open-source framework developed by Google that offers various pre-trained models for computer vision tasks such as object detection, hand tracking, and pose estimation.

For pose estimation, Mediapipe provides a pre-trained model that detects and tracks multiple human body keypoints in real-time. The model is based on the EfficientDet-D0 architecture and uses a top-down approach to predict 21 keypoints for each hand and 33 keypoints for each detected person, for the pose.

Figure 4.1: Visualization of keypoints extracting using Media Pipe

In the above figure the coordinates of every frames of a input video are obtained from mediapipe keypoints generation method and are plotted on the video to get the visualisations. The keypoints for left hand, right hand and pose are collected and marked in blue shades. The blue area is considered as the region of interest during training of model.

Algorithm 1 shows the implementation details for generating keypoints for hands using Mediapipe hand tracking(OpenCV). Same algorithm is used to generate keypoints for the pose.

Algorithm 1 Generating Hand Keypoint

Require: List of file paths to static frames of videos.

1: Import the necessary libraries: OpenCV, MediaPipe.
2: Define constants Hand1 and Hand2 for MediaPipe hand tracking.
3: **for** for every filepath in the list **do**
4: Read ith frame using OpenCV and flip it along the y-axis
5: Convert ith image from BGR to RGB and process it with MediaPipe's hand tracking.
6: Append the (x,y) coordinates of both hands in ith frame to Hand1X, Hand1Y, Hand2X, Hand2Y
7: **end for**
8: **return** $(Hand1X, Hand1Y, Hand2X, Hand2Y)$

4.2 Extracting CNN features

Once the keypoints are generated these key points are feeded to CNN network. MobileNetV2 is a convolutional neural network architecture that was designed for feature extractor for transfer learning tasks and also for the efficient image classification tasks.In transfer learning, a pre-trained neural network is used as a feature extractor, and its learned features are used to train a new neural network for specific task.Feature extraction is a crucial step in many machine learning applications,it is a component that takes input data and transforms it into a set of features. The "100" in the model's name refers to the width multiplier used in the architecture. Specifically, the width multiplier is set to 1.0 in the MobileNetV2-100 model, which means that the number of filters in each layer is the same as in the original MobileNetV2 architecture. Algorithm 2 shows the implementation of extracting the CNN features from keypoints generated uisng Mediapipe.

Algorithm 2 Extracting CNN features

Require: Keypoints of hand1, hand2 and pose

1: Let `frame_length` and `frame_width` be the height and width of the output image frames.
2: Initialize an instance of CNN.
3: Initialize an empty array to store the features.
4: **for** Each frame in the video **do**
5: Create an image array of shape (`frame_length`, `frame_width`, 3) and fill it with zeros.
6: Convert image to float32. Resize image to (224, 224). Call the function bodypose() to read every keypoints of i^{th} frame
7: Append feat to features
8: **end forreturn** features extracted

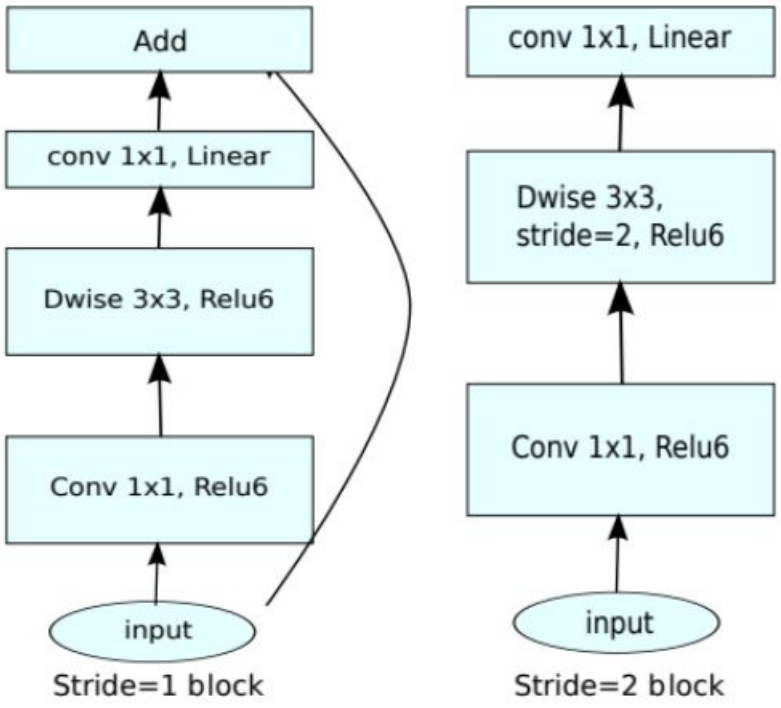

Figure 4.2: MobileNetV2 convolutional neural network architecture

4.3 Augmentation of CNN features

An imbalanced dataset can lead to biased model training and affect the accuracy and effectiveness of the machine learning models. When a dataset is imbalanced, a model may tend to predict the majority class more often, leading to poor performance on the minority class. This is because the model is more likely to be biased towards the majority class due to the larger number of samples. As a result, the model's overall accuracy may be high, but it may perform poorly on the minority class during testing. So it is necessary to have a balanced dataset or use some augmentation methods to mitigate effects of imbalanced data.

Augmentation can be made either to the videos using the techniques of undersampling and oversampling or to the data features.

SMOTE (Synthetic Minority Over-sampling Technique) is a popular data augmentation technique used to balance class distributions in imbalanced datasets. In the context of CNN feature augmentation, SMOTE can be used to increase the size of the dataset by creating synthetic samples of the minority class, which can help improve model performance by reducing overfitting and increasing the representativeness of the dataset.

Algorithm 3 shows the implementation details for augmenting CNN features using SMOTE method.

Algorithm 3 Augmentation of CNN features

Require: CNN features extracted from keypoints.

1: Initialize an empty numpy array to store the augmented features.
2: Create a SMOTE object with the desired parameters.
3: Apply the SMOTE algorithm to the original features to generate synthetic samples.
4: Add the original and synthetic features to the numpy array
5: Shuffle the numpy array to ensure the augmented dataset is random.
6: **return** (augmented features)

4.4 Attention based BiLSTM Model

The BiLSTM design is a kind of recurrent neural network that analyses the input sequence in both forward and backward directions, enabling the network to recognise relationships in the sequence. This is better than a unidirectional BiLSTM.

Based on their importance to the current output, the attention mechanism is utilised to focus on particular segments of the input sequence. This is accomplished by calculating a set of attention weights, which represent the proportions of each input sequence component that go into the final output. These weights are calculated using a different neural network, often a feedforward network, which receives as input the BiLSTM's current hidden state.

For a variety of natural language processing tasks, it has been shown that the combination of BiLSTM with attention is quite successful. By concentrating on the most crucial segments of a sequence for the present output, it enables the network to overcome long-range dependencies in the input sequence. Additionally, the attention mechanism offers a natural approach to understand the model's output by highlighting the elements of the input sequence that were crucial for the prediction.

After augmentation we obtain the fixed size feature embeddings. A BiLSTM with 5 layers is used to process these embeddings and generate class labels. The output is flatten to obtain 2-dimensional tensor and classified into number of classes present in the dataset i,e,. 113. An earlystopping with patience of 15 epochs is set to prevent overfitting and improve generalisation of the model. The model was trained with Adam optimiser, batch size of 32, learning rate of 0.0001 for 100 epochs.

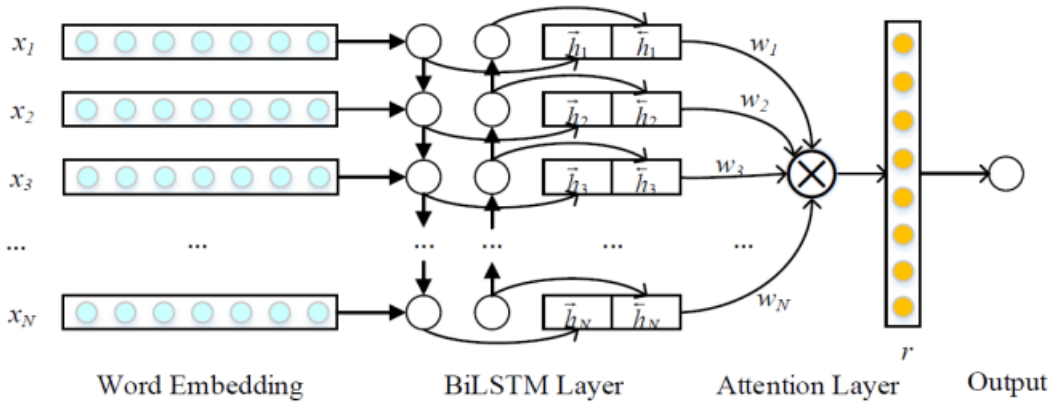

Figure 4.3: Basic structure of BiLSTM model with Attention

Chapter 5

RESULTS AND DISCUSSIONS

5.1 Quantitative Analysis

In this section we provide a quantitative analysis of different methods proposed on our dataset and INCLUDE-50 dataset. We report the accuracies of models with these methods. We also provide qualitative analysis of our dataset.

5.1.1 Analysis of training and validation accuracy

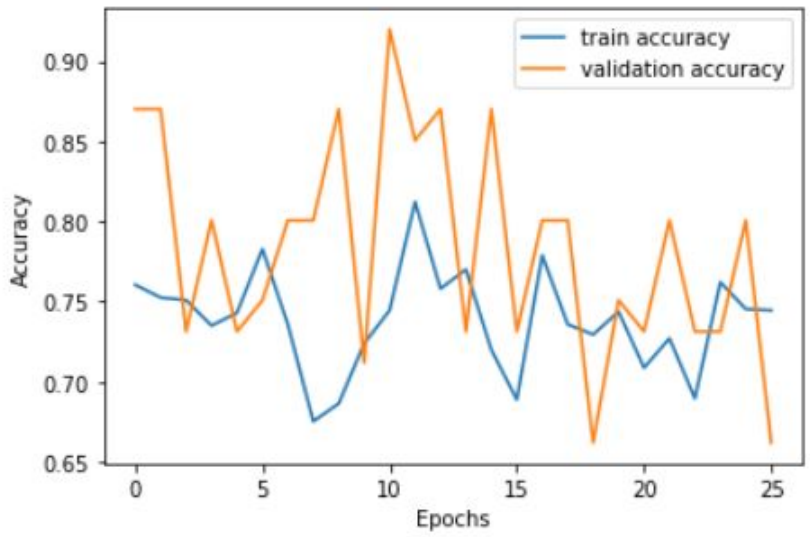

Figure 5.1: Representation of Train and Validation Accuracies

The plot depicts the learning of the model for train set and validation set. The model showed higher accuracy for validation data than training data. The presence of more representative real world samples in validation set, or the presence of less augmented features in validation set leading to more generalisation could be the probable reason for this result. The train accuracy at the 26th epoch is almost the same as the initial epoch with variations in between. The validation accuracy decreased from to 87 to 66 with 93 being the maximum accuracy.

5.2 Comparison of Datasets

OUR Dataset

Characteristic	Data
Categories	67
Word	36
Sentence	16
Character	61
Videos	388
Frame Rate	30fps
Resolution	1080 x 1920

INCLUDE-50 Dataset

Characteristic	INCLUDE-50
Categories	15
Words	50
Videos	958
Frame Rate	25fps
Resolution	1920x1080

Figure 5.2: Comparision of Dataset

5.2.1 Accuracies and Result comparison

We combined the data at character level and word level to make a good number of videos for training the model. The accuracies obtained with and without augmentation is shown in Table 5.1.

Table 5.1: Results obtained for our dataset combining character and word level videos

Model	Feature Augmentation	Pose Estimation	Feature Extraction	Accuracy(in %)
Baseline (BiLSTM)	No	Media pipe	Mobile Net V2	73.87
Attention based BiLSTM	Yes	Media pipe	Mobile Net V2	81.69

We also evaluated the model with INCLUDE-50 dataset. INCLUDE-50 dataset that we obtained was highly imbalanced. The modified model could produce balanced features as input to the network and we obtained an increase in accuracy to a great extent when compared to

their baseline model without augmentation as shown in Table 5.2. The modification that we employed thus can be implemented generally for any dataset. We extended our experiment

Table 5.2: Results obtained for INCLUDE-50 dataset

Model	Feature Augmentation	Pose Estimation	Feature Extraction	Accuracy(in %)
Baseline (BiLSTM) [9]	No	Openpose	Mobile Net V2	73.9
Attention based BiLSTM	Yes	Media pipe	Mobile Net V2	93.42

separately on character level and word level data. Total number of videos in character level is 479 and in word level is 199 videos. A pretrained model trained on INCLUDE-50 dataset was used to evaluate these data.

Table 5.3: Results obtained for our dataset combining character and word level videos

Dataset level	Model	Feature Augmentation	Pose Estimation	Feature Extraction	Accuracy(in %)
Character	Baseline (BiLSTM)	No	Openpose	Mobile Net V2	73.9
Character	Attention based BiLSTM	Yes	Media pipe	Mobile Net V2	93.42
Word	Baseline (BiLSTM)	No	Openpose	Mobile Net V2	52.27
Word	Attention based BiLSTM	Yes	Media pipe	Mobile Net V2	71.53

5.3 Qualitative Analysis

The model is tested by giving video clips of classes that are not present in the training and validation set to check the generalisation of the model. The model could predict some of the classes precisely but failed for few others completely. It predicted correctly for the data with sufficient augmentation of features. For the videos with less augmentation, the model fails to predict the correct kannada letters or mathra. Few positive results and negative results of the sample input videos from test set is shown below.

5.3.1 Positive Results

Ground Truth Text:ಒಂದು
Predicted Text: ಒಂದು

Ground Truth Text:ಐಳು
Predicted Text: ಐಳು

Ground Truth Text:ಬೇಸಿಗೆ
Predicted Text: ಬೇಸಿಗೆ

Ground Truth Text:ನೀವು
Predicted Text: ನೀವು

5.3.2 Negative Results

Ground Truth Text:ನಿನ್ನೆ
Predicted Text: ನಿನ್ನ

Ground Truth Text:ನೀಲಿ
Predicted Text: ಸಲ

Chapter 6

CONCLUSION AND FUTURE SCOPE

In this work, we translate Indian Sign Language into regional language(kannada). We created a sign language dataset at the character, word, and sentence levels in the regional kannada language and tested it at the character and word levels. We conducted a comparison with a base model, whose accuracy was 73.87 %. Our model has an accuracy of 81.69%. Additionally, we evaluated our model to the INCLUDE-50 dataset and found that it had a greater accuracy of 93.42 percent.

With an increase in the model's performance, this work can be expanded to create more classes and be implemented at the sentence level.

REFERENCES

[1] Ramesh Mahadev Kagalkar and Shyamrao V Gumaste. Mobile application based translation of sign language to text description in kannada language. volume 12, pages 92–112, 2018.

[2] Yanwei Pang, Yuan Yuan, Xuelong Li, and Jing Pan. Efficient hog human detection. *Signal processing*, 91(4):773–781, 2011.

[3] SVM Vishwanathan and M Narasimha Murty. Ssvm: a simple svm algorithm. In *Proceedings of the 2002 International Joint Conference on Neural Networks. IJCNN'02 (Cat. No. 02CH37290)*, volume 3, pages 2393–2398. IEEE, 2002.

[4] Ramesh Chandra Poonia. List: A lightweight framework for continuous indian sign language translation. volume 14, page 79. MDPI, 2023.

[5] Yong Yu, Xiaosheng Si, Changhua Hu, and Jianxun Zhang. A review of recurrent neural networks: Lstm cells and network architectures. *Neural computation*, 31(7):1235–1270, 2019.

[6] Rahaf Abdulaziz Alawwad, Ouiem Bchir, and Mohamed Maher Ben Ismail. Arabic sign language recognition using faster r-cnn. volume 12. Science and Information (SAI) Organization Limited, 2021.

[7] Tiantian Yuan, Shagan Sah, Tejaswini Ananthanarayana, Chi Zhang, Aneesh Bhat, Sahaj Gandhi, and Raymond Ptucha. Large scale sign language interpretation. In *2019 14th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2019)*, pages 1–5. IEEE, 2019.

[8] Ramesh M Kagalkar and SV Gumaste. Curvilinear tracing approach for extracting kannada word sign symbol from sign video. *International Journal of Image, Graphics and Signal Processing*, 9(9):18, 2017.

[9] Advaith Sridhar, Rohith Gandhi Ganesan, Pratyush Kumar, and Mitesh Khapra. Include: A large scale dataset for indian sign language recognition. In *Proceedings of the 28th ACM international conference on multimedia*, pages 1366–1375, 2020.

[10] Mahmoud M Zaki and Samir I Shaheen. Sign language recognition using a combination of new vision based features. *Pattern Recognition Letters*, 32(4):572–577, 2011.

[11] 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 *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 1459–1469, 2020.

[12] Jens Forster, Christoph Schmidt, Oscar Koller, Martin Bellgardt, and Hermann Ney. Extensions of the sign language recognition and translation corpus rwth-phoenix-weather. In *LREC*, pages 1911–1916, 2014.

[13] Marie Alaghband, Niloofar Yousefi, and Ivan Garibay. Facial expression phoenix (feph): an annotated sequenced dataset for facial and emotion-specified expressions in sign language. *arXiv preprint arXiv:2003.08759*, 2020.

[14] Sang-Ki Ko, Jae Gi Son, and Hyedong Jung. Sign language recognition with recurrent neural network using human keypoint detection. In *Proceedings of the 2018 conference on research in adaptive and convergent systems*, pages 326–328, 2018.

[15] Sang-Ki Ko, Chang Jo Kim, Hyedong Jung, and Choongsang Cho. Neural sign language translation based on human keypoint estimation. *Applied sciences*, 9(13):2683, 2019.

[16] Sutanu Nandi, Abhishek Subramanian, and Ram Rup Sarkar. An integrative machine learning strategy for improved prediction of essential genes in escherichia coli metabolism using flux-coupled features. *Molecular BioSystems*, 13(8):1584–1596, 2017.

[17] Amirsaeed Yazdani, Sumit Agrawal, Kerrick Johnstonbaugh, Sri-Rajasekhar Kothapalli, and Vishal Monga. Simultaneous denoising and localization network for photoacoustic target localization. *IEEE transactions on medical imaging*, 40(9):2367–2379, 2021.

[18] Karen Emmorey and David Corina. Lexical recognition in sign language: Effects of phonetic structure and morphology. *Perceptual and motor skills*, 71(3_suppl):1227–1252, 1990.

[19] 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, 2008.

[20] P Kishore, S Jyothi, Ghouse Basha, SVB Rao, M Rajeevan, Isabella Velicogna, and Tyler C Sutterley. Precipitation climatology over india: validation with observations and reanalysis datasets and spatial trends. *Climate dynamics*, 46:541–556, 2016.

Appendix A

A.1 Smote method for feature Augmentation

Uses SMOTE-E algorithm to augment the CNN features of minority class in a video dataset.

Parameters:

features: numpy array of shape (n_samples, n_features_per_sample) Features of video samples.

labels: numpy array of shape (n_samples,) Labels of video samples.

n_samples: int Number of synthetic samples to generate for each class (default: 1000).

Returns:

synthetic_features: numpy array of shape (n_samples * 2, n_features_per_sample) Synthetic features generated by SMOTE-E.

synthetic_labels: numpy array of shape (n_samples * 2,) Synthetic labels generated by SMOTE-E.

```python
from imblearn.over_sampling import SMOTE

def smote_e_video_augmentation(features, labels, n_samples=1000):

    # Create SMOTE object
    smote = SMOTE(sampling_strategy='auto', k_neighbors=5, n_jobs=-1)

    # Fit SMOTE to features and labels
    synthetic_features, synthetic_labels = smote.fit_resample(features, labels)

    # Generate synthetic samples for each class
    minority_indices = np.where(synthetic_labels == 1)[0]
    majority_indices = np.where(synthetic_labels == 0)[0]

    minority_synthetic_indices = np.random.choice(minority_indices, n_samples)
    majority_synthetic_indices = np.random.choice(majority_indices, n_samples)

    synthetic_indices = np.concatenate([minority_synthetic_indices, majority_synthetic_indices])
    synthetic_features = synthetic_features[synthetic_indices]
    synthetic_labels = synthetic_labels[synthetic_indices]

    return synthetic_features, synthetic_labels
```

Figure A.1: code snippet of smote feature augmentation method