TRIBHUVAN UNIVERSITY

INSTITUTE OF ENGINEERING

PULCHOWK CAMPUS

PROJECT REPORT

ON

GESTURE SYNTHESIS USING MULTIMODAL SUPERVISED LEARNING

SUBMITTED BY:

PRASUN BHANDARI (PUL075BEI025)

RAHUL ROUNIYAR (PUL075BEI027)

RONAB SHRESTHA (PUL075BEI029)

SUNIL GURAU (PUL075BEI043)

SUBMITTED TO:

DEPARTMENT OF ELECTRONICS & COMPUTER ENGINEERING

May, 2023

Page of Approval

TRIBHUVAN UNIVERSIY
INSTITUTE OF ENGINEERING
PULCHOWK CAMPUS
DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING

The undersigned certifies that they have read and recommended to the Institute of Engineering for acceptance of a project report entitled **"Gesture Synthesis Using Multimodal Supervised Learning"** submitted by **Prasun Bhandari**, **Rahul Rouniyar**, **Ronab Shrestha**, **Sunil Gurau** in partial fulfillment of the requirements for the Bachelor's degree in Electronics & Computer Engineering.

.....
Supervisor
Asst. Prof. Jitendra Kumar Manandhar
Department of Electronics and Computer Engineering,
Pulchowk Campus, IOE, TU.

.....
Internal examiner

Department of Electronics and Computer Engineering,
Pulchowk Campus, IOE, TU.

.....
External examiner

Date of approval:

Copyright

Acknowledgments

We are grateful to the Department of Electronics and Computer Engineering for incorporating the major project as a part of our engineering course. This project surely will enhance our understanding of concepts we have learnt so far. We are also grateful to **Asst. Prof. Jitendra Kumar Manandhar** for his constant support, guidance and motivation throughout the project. We also would like to express our heartfelt gratitude towards Institute of Engineering that circumvented the difficulty imposed by this pandemic on holding physical classes, by conducting online classes and not letting the academic progress of students grind to a halt. Also, we are thankful to several of our teachers and mentors like Nanda Bikram Adhikari, Lok Nath Regmi, Ram Krishna Maharjan who have been closely inspecting and monitoring our project while providing necessary guidelines and support alongside. We would also like to acknowledge the love and support of our family and friends who are always there for our support.

Abstract

One of the long-standing ambitions of the modern science and engineering has been to create a non-human entity that manifests human-like intelligence and behavior. One step to achieving the goal is executing a communication just like the humans do. Human speech is often accompanied by a variety of gestures which add rich non-verbal information to the message the speaker is trying to convey. Gestures add clarity to the intention and emotions of the speaker and enhance the speech by adding visual cues alongside audio signal. Our project aims to synthesize co-speech gestures by learning from individual speaker's style. We follow a data-driven approach instead of rule-based approach as the audio-gesture relation is poorly captured by a rule-based system due to issues like asynchrony and multi-modality. As is the current trend, we train the modal from in-the-wild videos embedded with audio instead of relying on the motion capture of subjects in lab for video annotation. For establishing the ground truth for the data set of video frames, we rely on an automatic pose detection system. Although the ground truth signal tends to be not as accurate as manually annotated frames, the approach relieves us of time and labor expense. We perform the cross-modal translation from monologue speech of a single speaker to their hand and arm motion based on the learning of temporal correlation between the sequence of pose and audio sample.

Keywords: *Gesture synthesis, Supervised learning, Human Computer Interaction, Multi-modality, Pose Estimation, Temporal Context*

Contents

List of Figures

List of Abbreviations

HCI	Human Computer Interaction
PCK	Percent of Correct Keypoints
FTD	Frechet Template Distance
ECA	Embedded Conversational Agent
GAN	Generative Adversial Network
VAE	Variational Auto Encoder
MFCC	Mel-Frequency Cepstral Coefficients
GRU	Gated Recurrent Unit
RNN	Recurrent Neural Network
ANN	Artificial Neural Network
NN	Neural Network
LSTM	Long-short Term Memory
BPTT	Backpropagation Through Time
CNN	Convolution Neural Network
CVC	Computer Vision Center
COCO	Common Objects in Context
MAP	Mean Average Position

1. Introduction

1.1 Background

Science today is obsessed with the idea of making a replica of a human that has human-like intelligence, behavior and morphology. There are many researches and projects being conducted to make the idea a reality. Some of these include lip-syncing, face swapping, pose-retargeting at which we achieved success to a major extent. However, the task of co-speech gesture synthesis is a challenging one as it must deal with the challenge of understanding the inherent human cognition.

When we talk, our voice is most often accompanied by hand and arm gestures thus augmenting our audio channel of communication with a visual channel. Although most of the information a speaker is trying to convey is provided by the speech, the visual information delivered by the gestures about the emotion, attitude and intention of the speaker cannot be disregarded. Gestures allow individuals to communicate a variety of feelings and thoughts, from contempt and hostility to approval and affection, often together with body language in addition to words when they speak. Gesticulation and speech work independently of each other but join to provide emphasis and meaning.

The tendency to gesture in humans in communication can be attributed to evolution. In ancient times, when no verbal media of communication had originated, archaic humans solely relied on the visual medium for communicating. Decoding gestures from fellow humans was vital to survival when hunting in bands or being attacked by some wild beast. When the languages developed, although humans no longer needed gesticulation for communication, the evolutionary hysteresis caused the upcoming generations to still use gestures as a mechanism of communication.

In other words, Gesture synthesis refers to the generation of natural and expressive body movements or gestures by computational methods. It involves developing algorithms and models that can synthesize realistic and meaningful gestures that can be used for a variety of applications, such as animation, virtual reality, robotics, and human-computer interaction.

Gesture synthesis can be achieved using different techniques, including rule-based methods, data-driven methods, and hybrid approaches. Rule-based methods involve defining a set of rules or constraints that govern the generation of gestures based on certain parameters, such as the context or the intended meaning. Data-driven methods, on the other hand, involve training models on large datasets of motion capture data or videos of human movements to

learn patterns and generate new gestures.

Recent advances in machine learning and deep learning have led to the development of sophisticated models for gesture synthesis, such as generative adversarial networks (GANs), variational auto-encoders (VAEs), and recurrent neural networks (RNNs). These models can generate highly realistic and natural-looking gestures that capture the nuances and intricacies of human movements.

Applications of gesture synthesis include virtual characters and avatars that can communicate with humans using gestures, robots that can perform tasks more naturally and efficiently, and interactive systems that can recognize and respond to human gestures in real time. Gesture synthesis has the potential to revolutionize the way we interact with machines and create more immersive and engaging experiences for users.

1.2 Motivation

Our decision to focus on gesture synthesis for our major project was driven by several key factors. Firstly, gesture synthesis has the potential to create more immersive and interactive experiences in fields such as virtual reality and gaming, enabling users to control virtual environments and objects using natural and expressive body movements. Secondly, gesture synthesis can automate the animation process, reducing the time and effort required to create complex and nuanced movements for characters or other virtual objects, leading to increased efficiency. Thirdly, gesture synthesis can improve human-robot interaction by enabling more effective and natural communication through intuitive body language. Fourthly, gesture synthesis can enhance accessibility for individuals with physical disabilities, enabling them to control virtual environments, communicate more effectively, or participate in physical therapy and rehabilitation. Fifthly, gesture synthesis can also improve sign language recognition systems, allowing for more natural and expressive communication between individuals who use sign language and those who do not. Lastly, gesture synthesis can be used to study human movement and biomechanics, providing insights into the mechanics of natural and expressive body movements. While the motivations for gesture synthesis vary depending on the specific application, they are generally focused on improving the efficiency, effectiveness, and naturalness of human-computer or human-robot interactions, as well as enhancing accessibility and understanding of human movement.

1.3 Problem statements

Gesture synthesis is the process of generating natural and expressive body movements, such as hand gestures or full-body poses, that can be used in a variety of applications, such as virtual reality, animation, and robotics. This involves designing algorithms that can

automatically generate these movements, either based on input from a human operator or from an artificial intelligence system. However, gesture synthesis poses a number of challenges that must be overcome to achieve this goal. One of the primary challenges is creating natural and expressive body movements that can convey a range of emotions and meanings. Additionally, generating movements that are physically plausible and realistic, while taking into account the biomechanical constraints of the human body, is another major challenge. Gesture synthesis often requires a large amount of training data, which can be difficult and expensive to obtain, and the quality of the data can impact the quality of the synthesized gestures. There is also a wide range of variation in how different people perform the same gestures, which makes it challenging to generalize across individuals. Furthermore, gesture synthesis often involves modeling complex interactions between different body parts, which can be computationally intensive. Generating coordinated and well-timed movements is another challenge, particularly when trying to synchronize movements with other actions or events. Finally, gestures can take many different forms, including hand movements, facial expressions, and body posture, making it challenging to synthesize a wide range of multimodal gestures in a coherent and natural way. These challenges highlight the need for sophisticated algorithms that can handle complex and nuanced movements while taking into account the physical and cognitive constraints of the human body. Overall, the goal of gesture synthesis is to create algorithms that can generate natural and expressive body movements in a way that is both efficient and effective, and that can be easily integrated into a variety of applications.

1.4 Objectives

The main objectives of our project are as follows:

- Create an end-to-end model that maps audio to a gesture sequence
- Learn individual styles of gesticulation
- Extend the work in data-driven approach to gesture synthesis

1.5 Scope

Gesture synthesis has a wide range of potential applications in various fields. One such application is in virtual reality, where gesture synthesis can be used to create more immersive experiences by allowing users to interact with virtual objects and environments using natural and expressive body movements. In animation, gesture synthesis can automate the animation process, freeing up animators to focus on more creative aspects of the animation. Gesture synthesis can also be used in robotics to program robots to perform complex movements,

which enables them to interact more effectively with humans in various settings. In gaming, gesture synthesis can create more interactive experiences by allowing players to control game characters using natural and intuitive body movements. In healthcare, gesture synthesis can be used in physical therapy and rehabilitation to help patients regain movement and strength in injured limbs or to monitor and track treatment progress. Finally, gesture synthesis can be used to develop more natural and expressive sign language recognition systems, allowing deaf or hard-of-hearing individuals to communicate more effectively with others. These examples illustrate the vast potential of gesture synthesis across different fields and industries.

2. Literature Review

2.1 Conversational gestures

There are basically four kinds of categorizations of co-speech gestures: iconic gestures, metaphorical gestures, beat gestures, and deictic gestures. Iconic and metaphorical gestures both carry meaning and are used to visually enrich our communication[1]. An iconic gesture can be an up and down movement to indicate, for example, the action of slicing a tomato. Instead, a metaphoric gesture can involve an empty palm hand that is used to symbolize 'presenting a problem'. In other words, metaphoric gestures have an arbitrary relation to the concept they communicate, and iconic gestures have a form that is visually related to the concept being communicated. Iconic and metaphoric gestures not only differ in terms of content and presentation but are also processed differently in the brain. Beat gestures do not carry semantic meaning, and they are often used to emphasize the rhythm of speech. Beat gestures have been shown to both facilitate speech and word recall and are the most frequent type of gesture. Finally, deictic gestures are used to point out elements of interest or to communicate directions. Not only do they enhance spoken communication, they also facilitate learning.

Regarding the origin of speech and gesture in reference to one another, it is suggested that gesture and speech originate from a common source and thus should co-occur in time according to well-defined rules[1]. Although, it is also found in some research that gesture starts before the corresponding utterance[2]. Others even contend that there is still uncertainty regarding the temporal links between speech and gesture and that a gesture may come prior to, following, or in the middle of an utterance.

2.2 Co-speech gesture synthesis

Synthesizing co-speech gestures has been an active topic in robotics, graphics, and vision. A recent trend in this task is using in-the-wild videos rather than those collected in lab scenarios with sensors, extending the variety of the synthesized gestures. The ambiguity of the job, however, results in the under-fitting of the data and lack of expressiveness of the outputs, which is a hurdle in the path of genuine co-speech gesture production[3]. Although Ginosar et al.[3] implemented adversarial learning to improve gesture quality, the model still significantly relies on the regression loss to generate synchronized motions with the audio, resulting in a deterministic, monotonous outcome. By enclosing each gesture in a shared style space across

subjects and accomplishing style transfer or preservation by changing the style embedding, it is found that it is possible to untangle the style and content of gestures[4]. With only one typical gesture for each subject, the styles are specified separately for each subject. The probabilistic model MoGlow, based on normalizing flows [5], is introduced by Alexanderson et al. [6] to represent the mapping from gestures to Gaussian distributions conditional on the input audio. This model can accurately represent the one-to-many mapping by sampling latent vectors from Gaussian distributions during inference. The normalizing flows model [5] only enables linear operations, which restricts the model's expressiveness. With template vector learning, our model eliminates the ambiguity of one-to-many mapping and achieves diverse generations by sampling the template vector during inference. A more recent approach by Qian et al. [7] relieves the ambiguity of one-to-many mapping template vector learning and accomplishes diverse generation by sampling the template vector when inferencing.

2.3 Evaluation metric

The regression loss (L1) has been found to be used the most for the evaluation of the accuracy of the synthesized gestures [3] [8]. L1 Loss Function is used to minimize the error which is the sum of the all the absolute differences between the true value and the predicted value. There are works done in this regard using the PCK where, if a predicted keypoint is within max(h, w) pixels of the ground truth keypoint, where 'h' and 'w' are the height and breadth of the person bounding box, respectively, then it is considered to be correct. MG (Mo-Glow) system without style control, has found to be used where the main metrics found to be used are MG-H (hand height control), MG-V (velocity control), MG-R (gesture radius control) and MG-S (gesture symmetry control) [6]. Basically, most of the methods that have used quantitative evaluation metrics, have used a baseline model to compare with the synthesized model and checked for the correctness using fundamentally the aforementioned techniques. Although few qualitative methods have also been used they are not that signigicant in determining the success of the synthesization. Different variations of Frechet distance, particularly FTD, have also been used as an evaluation metric in similar works as variety is discouraged when a generated gesture sequence's distance from reality is measured directly [7].

3. Theoretical Background

3.1 Multimodal machine learning

Multimodal machine learning is a subfield of artificial intelligence and machine learning that focuses on developing algorithms and models that can analyze and interpret data from multiple modalities, such as text, images, video, audio, and sensor data. The goal of multimodal machine learning is to combine information from different modalities to create a more complete and accurate representation of the data. For example, combining text and image data to better understand the content of a social media post, or using audio and sensor data to detect anomalies in industrial equipment. Multimodal machine learning algorithms can be used for a wide range of applications, including natural language processing, computer vision, speech recognition, and robotics. One of the main challenges in multimodal machine learning is how to effectively integrate information from different modalities, as each modality may have different features, formats, and levels of noise.
Recent advances in deep learning, neural networks, and reinforcement learning have greatly improved the performance of multimodal machine learning models, making it possible to achieve state-of-the-art results in many tasks. As the amount and variety of multimodal data continues to grow, multimodal machine learning is expected to play an increasingly important role in solving complex problems and developing more advanced AI systems. It involves the following major challenges[9]:

1. Representation: A first fundamental challenge is learning how to represent and summarize multimodal data in a way that exploits the complementarity and redundancy of multiple modalities. The heterogeneity of multimodal data makes it challenging to construct such representations. For example, language is often symbolic while audio and visual modalities will be represented as signals.

2. Translation: A second challenge addresses how to translate (map) data from one modality to another. Not only is the data heterogeneous, but the relationship between modalities is often open-ended or subjective. For example, there exist a number of correct ways to describe an image and and one perfect translation may not exist.

3. Alignment: A third challenge is to identify the direct relations between (sub)elements from two or more different modalities. For example, we may want to align the steps

in a recipe to a video showing the dish being made. To tackle this challenge, we need to measure similarity between different modalities and deal with possible long-range dependencies and ambiguities.

4. Fusion: A fourth challenge is to join information from two or more modalities to perform a prediction. For example, for audio-visual speech recognition, the visual description of the lip motion is fused with the speech signal to predict spoken words. The information coming from different modalities may have varying predictive power and noise topology, with possibly missing data in at least one of the modalities.

5. Co-learning: A fifth challenge is to transfer knowledge between modalities, their representation, and their predictive models. This is exemplified by algorithms of co-training, conceptual grounding, and zero shot learning. Co-learning explores how knowledge learning from one modality can help a computational model trained on a different modality. This challenge is particularly relevant when one of the modalities has limited resources (e.g., annotated data).

3.2 MFCC

The Mel-Frequency Cepstral Coefficients (MFCCs) is a commonly used feature extraction technique in the field of speech recognition and signal processing. MFCCs are based on the human auditory system and are used to represent the spectral characteristics of a speech signal.

The Mel scale is a perceptual scale of pitches arranged in such a way that the distance between each pitch is perceived as equal by the human ear. MFCCs are based on the Mel scale, which maps the frequency range of a speech signal onto a non-linear frequency scale. The Mel filterbank is a set of overlapping triangular filters that are used to extract the spectral characteristics of a speech signal. Each filterbank is designed to capture the energy of a specific frequency range in the Mel-scale.

After the speech signal has been filtered through the Mel-filterbank, the resulting spectral energies are compressed using a type of transform called the Discrete Cosine Transform (DCT). The DCT reduces the dimensionality of the feature vector and decorrelates the filterbank energies.

The final output of the MFCC extraction process is a vector of MFCCs, which are the amplitudes of the DCT coefficients. MFCCs represent the spectral envelope of a speech signal and are commonly used as features for speech recognition systems.

In summary, MFCCs are a feature extraction technique that captures the spectral characteristics of a speech signal using a Mel-filterbank and DCT. MFCCs are commonly used as

features for speech recognition systems because they are effective in capturing the spectral envelope of a speech signal, which contains important information for identifying phonemes and words.

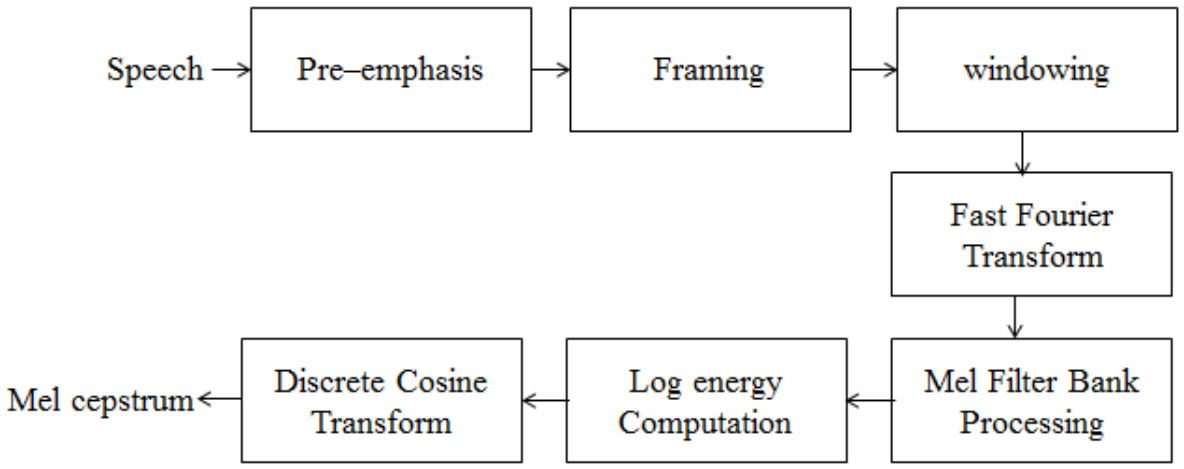


Figure 3.1: MFCC calculation.

3.3 Pose Estimation

Pose estimation refers to the process of estimating the position and orientation of objects in images or videos. In computer vision, pose estimation is often used to track human movement in applications such as motion capture, activity recognition, and virtual reality.

The first step in pose estimation is to detect salient features in the image or video. These features could be corners, edges, or other distinctive points that can be used to identify the object and track its movement over time.

Once the features have been detected, the next step is to match them across multiple frames in the video sequence. This is typically done by computing descriptors for each feature, such as SIFT, SURF, or ORB, and matching them based on their similarity. To estimate the 3D pose of an object from a 2D image, it is necessary to know the camera parameters, such as its intrinsic and extrinsic parameters. Intrinsic parameters include the focal length, principal point, and distortion coefficients, while extrinsic parameters describe the position and orientation of the camera relative to the scene.

There are several approaches to estimating the pose of an object, including model-based methods, feature-based methods, and deep learning-based methods. Model-based methods use a 3D model of the object to match against the image features, while feature-based methods estimate the pose directly from the image features. Deep learning-based methods use convolutional neural networks (CNNs) to directly regress the 2D or 3D pose of the object from the image.

The accuracy of pose estimation algorithms can be evaluated using various metrics, such as mean squared error, average angular error, or percentage of correct keypoints. Ground truth data, such as motion capture data or manual annotations, can be used to evaluate the performance of the algorithm.

Therefore, pose estimation involves detecting and matching salient features in images or videos, calibrating the camera parameters, and estimating the 3D pose of the object using various algorithms. Pose estimation is a fundamental problem in computer vision with many applications in areas such as robotics, augmented reality, and human-computer interaction.

3.3.1 Body-25 Model

Body-25 is a 2D human pose estimation model that can detect 25 key points on the human body, including joints, head, neck, shoulders, elbows, wrists, hips, knees, and ankles. The model is based on a convolutional neural network (CNN) architecture and is trained on a large dataset of labeled images.

The Body-25 model was introduced by researchers at the Computer Vision Center (CVC) at the Universitat Autònoma de Barcelona. It is an extension of the original Body-17 model, which can detect 17 key points in the human body.
The Body-25 model consists of several convolutional layers followed by max-pooling layers and fully connected layers. The input to the model is a grayscale image of a person, and the output is a set of 25 2D coordinates representing the location of the key points on the body.
The 25 key points detected by the Body-25 model are:
Head top, Neck, Right shoulder, Right elbow, Right wrist, Left shoulder, Left wrist, Right hip, Right knee, Right ankle, Left hip, Left knee, Left ankle, Chest, Right ear, Left ear, Right eye, Left eye, Right heel, Left heel, Right big toe, Left big toe, Right small toe, Left small toe.

Body keypoints are given as a sequence of 25 points and each of left hand keypoints and right hand keypoints are given as a sequence of 21 keypoints. The indexing of keypoints is based on a specific format. The body keypoints are in BODY_25 format. BODY_25 format is shown below:

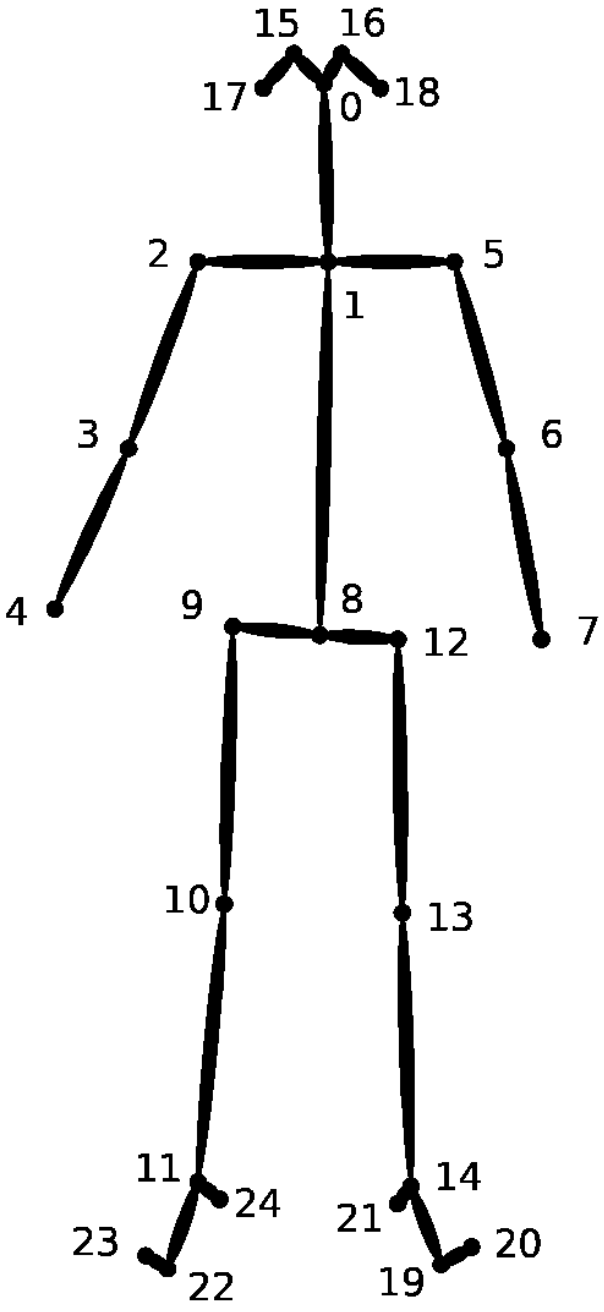


Figure 3.2: Body-25 Keypoints Format.

Left hand keypoints and right hand keypoints are not specified in the BODY-25 format. The format for these is shown in the figure below:

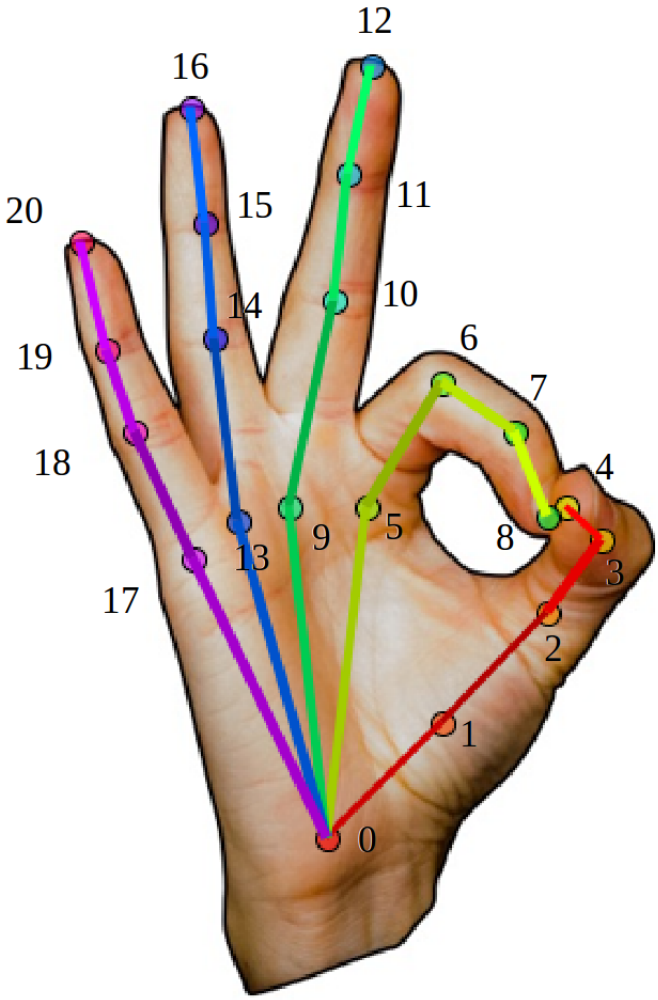


Figure 3.3: Hand Keypoints Format.

The Body-25 model has achieved state-of-the-art performance on several benchmarks, including the MPII Human Pose dataset and the COCO dataset. It is widely used in applications such as human-computer interaction, action recognition, and sports analytics.

3.3.2 COCO Model

The COCO model refers to a set of deep learning models that have been trained on the Common Objects in Context (COCO) dataset for object detection, segmentation, and other related tasks. The COCO dataset is a large-scale object detection, segmentation, and captioning dataset that contains over 330,000 images with more than 2.5 million object instances

labeled across 80 different object categories.

The COCO model is typically based on a CNN architecture, such as Faster R-CNN, Mask R-CNN, or YOLO. These models are trained to detect and classify objects in images, and in the case of Mask R-CNN, to also perform instance segmentation by predicting a binary mask for each detected object.

The COCO model is widely used in computer vision applications, such as autonomous driving, surveillance, and robotics. It has achieved state-of-the-art performance on several benchmarks, including the COCO dataset itself and the PASCAL VOC dataset.

The COCO model is also used as a backbone for many other object detection models that are fine-tuned on specific datasets or tasks. For example, the RetinaNet model uses a COCO pre-trained backbone and is fine-tuned on the PASCAL VOC dataset for object detection.

The COCO model has become a standard benchmark for evaluating object detection and segmentation models in computer vision research. The performance of the models is typically evaluated using mean average precision (MAP) and other related metrics on the COCO test set.

3.4 GAN

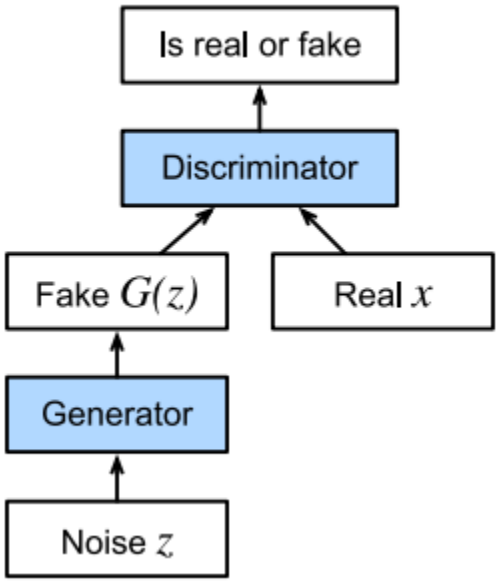


Figure 3.4: Architecture of GAN.

GAN stands for Generative Adversarial Network, which is a type of deep learning algorithm used in artificial intelligence and machine learning. GANs are composed of two neural networks, a generator and a discriminator, that work together to generate realistic outputs.

The generator generates synthetic data, such as images or sounds, by trying to mimic real data from a training set. The discriminator, on the other hand, tries to differentiate between the synthetic data and the real data. During training, the generator and discriminator play a game where the generator tries to fool the discriminator into thinking its synthetic data is real, while the discriminator tries to correctly identify the synthetic data as fake. Over time, the generator becomes better at generating realistic data, and the discriminator becomes better at identifying synthetic data, resulting in the production of more realistic and high-quality outputs.

3.5 Dense Layer

A dense layer is a fundamental building block of neural networks that is responsible for processing input data and producing output predictions. It consists of a set of neurons that are fully connected to the previous layer, allowing it to learn complex representations of the input data. The term "dense" refers to the fact that each neuron in the layer is connected to every neuron in the previous layer. During training, the weights and biases of the neurons in the dense layer are adjusted through back propagation, allowing the network to improve its predictions over time. The output of a dense layer can be further processed by additional layers or passed to the output layer for final predictions.

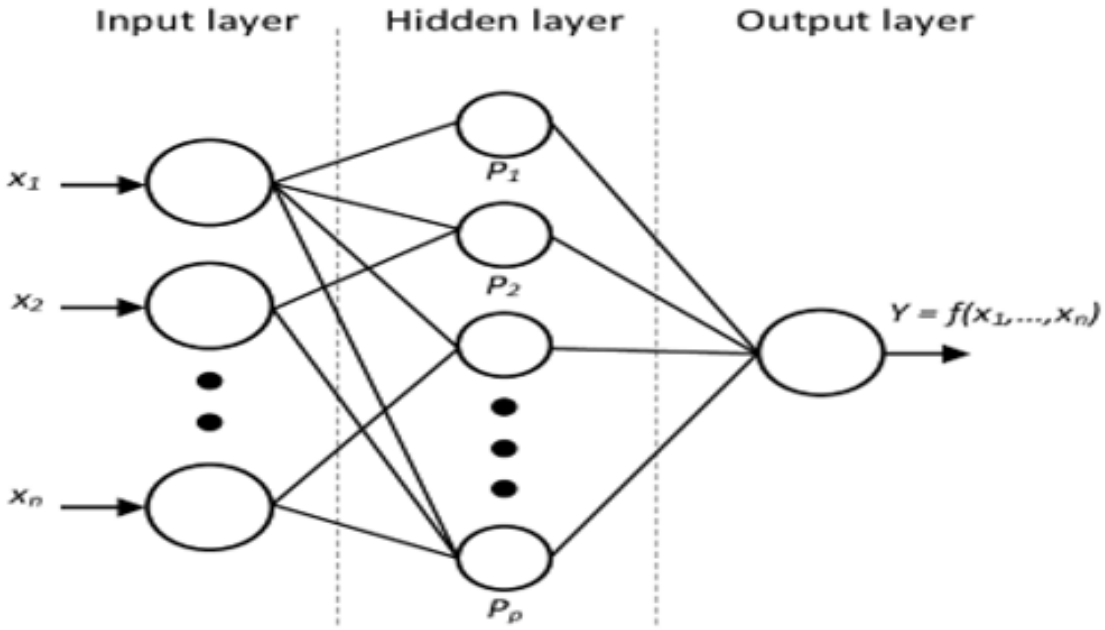


Figure 3.5: Architecture of Dense layer.

A dense layer, also known as a fully connected layer, is one of the most commonly used layer types in deep learning. It is typically used in the middle of a neural network, between

the input layer and output layer, to process and transform the input data.

Each neuron in a dense layer is connected to every neuron in the previous layer, which means that the layer has a large number of parameters that can be learned during training. This allows the dense layer to capture complex relationships between the input and output, making it well-suited for a wide range of tasks, such as image classification, natural language processing, and speech recognition.

3.6 Time Distributed Layer

The TimeDistributed layer in deep learning is a powerful tool for handling sequential data, such as videos, audio recordings, and time-series data. This layer applies a given neural network model to each time step of the input sequence independently, allowing the model to capture temporal dependencies and patterns in the data. In other words, it takes the output of one time step and feeds it as input to the next time step, thus enabling the model to learn from past events and predict future ones. The TimeDistributed layer is particularly useful in tasks such as speech recognition, sentiment analysis, and action recognition in videos, where understanding the temporal relationships between events is critical for accurate predictions. The TimeDistributed layer is a type of layer in recurrent neural networks (RNNs) and is

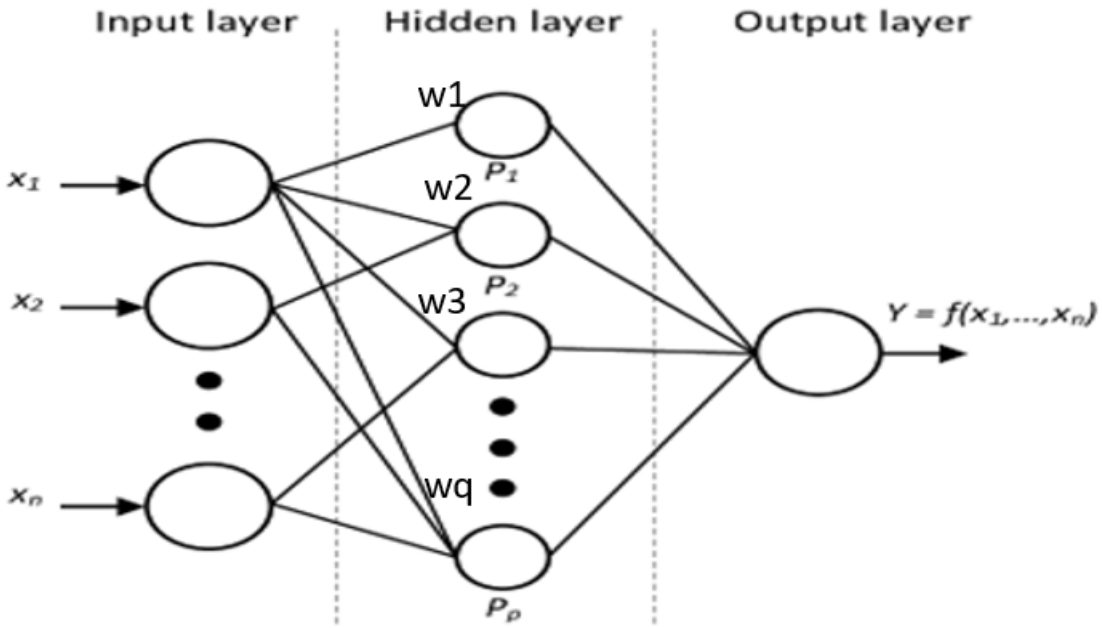


Figure 3.6: Architecture of Time Distributed Layer.

often used in combination with other layers, such as LSTM or GRU layers. It is designed to

handle sequential data with a fixed number of time steps, which can vary depending on the length of the input sequence.

3.7 GRU

Gated Recurrent Unit (GRU) is a type of RNN that was introduced in 2014 by Cho et al. GRU is similar to LSTM networks and is designed to address the vanishing gradient problem in RNNs.

RNNs are a type of neural network that is designed to handle sequential data such as time-series, natural language, and speech signals. In RNNs, each neuron receives an input and a hidden state from the previous time-step, and produces an output and a new hidden state that is passed to the next time-step.

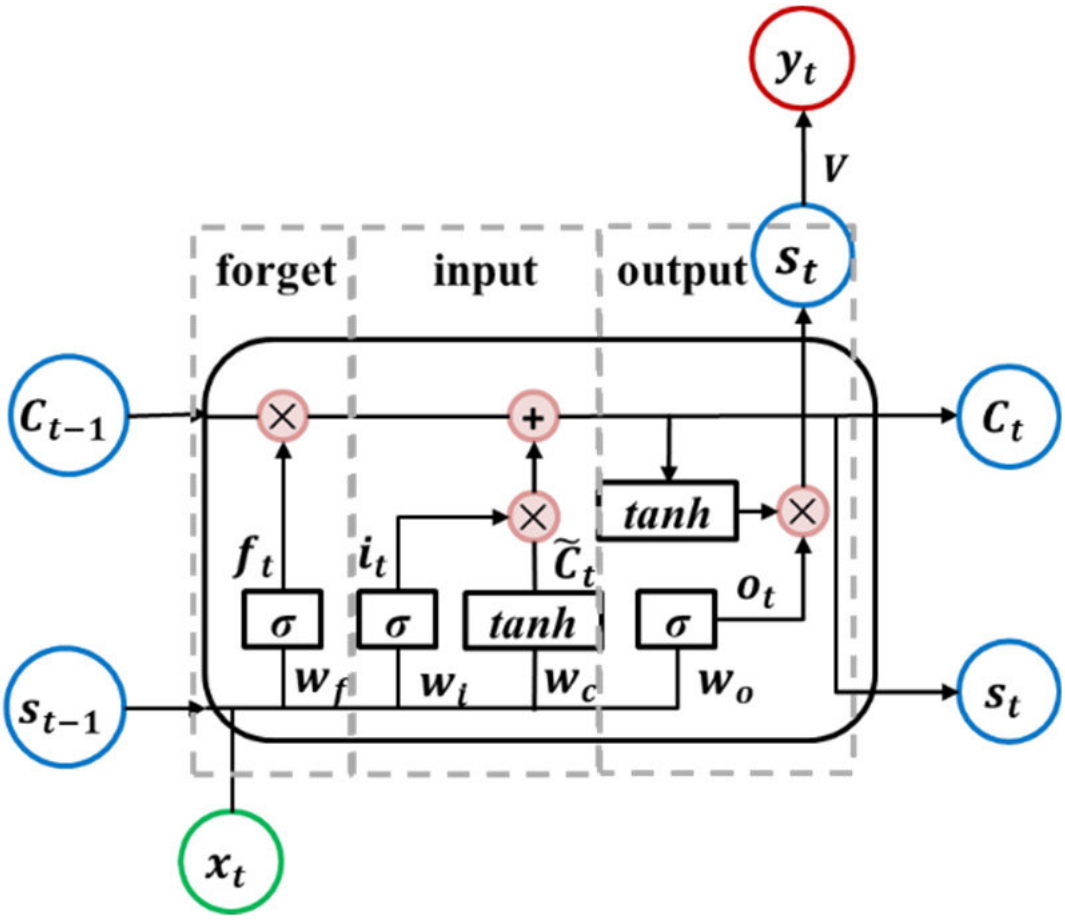


Figure 3.7: Architecture of GRU.

GRU is a variant of RNNs that has two gates, a reset gate and an update gate. The reset gate controls how much of the previous hidden state is combined with the current input, while the update gate controls how much of the new hidden state is kept and how much is updated.

Mathematically, the GRU update equations are as follows:

$$Reset\ gate : r\_t = sigmoid(W\_r * [h\_t - 1, x\_t]) \tag{3.1}$$

$$Update\ gate : z\_t = sigmoid(W\_z * [h\_t - 1, x\_t]) \tag{3.2}$$

$$Candidate\ hidden\ state : h'\_t = tanh(W\_h * [r_t * h\_t - 1, x\_t]) \tag{3.3}$$

$$Final\ hidden\ state : h\_t = (1 - z\_t) * h\_t - 1 + z\_t * h'\_t \tag{3.4}$$

where h_t is the hidden state at time t, x_t is the input at time t, r_t is the reset gate, z_t is the update gate, h'_t is the candidate hidden state, and W_r, W_z, and W_h are weight matrices that are learned during training. The reset gate allows the network to selectively forget or remember parts of the previous hidden state, while the update gate allows the network to update the hidden state based on the current input. The candidate hidden state is then computed by combining the previous hidden state and the current input using the reset gate, and this candidate hidden state is combined with the previous hidden state using the update gate to produce the final hidden state.

GRUs are computationally efficient and can be trained using BPTT. They have been shown to outperform traditional RNNs on a variety of sequence modeling tasks, including speech recognition, language modeling, and machine translation.

In summary, GRU is a type of RNN that uses two gates, a reset gate and an update gate, to selectively forget or remember parts of the previous hidden state and update the hidden state based on the current input. GRUs have several advantages over traditional RNNs and have been shown to be effective on a variety of sequence modeling tasks.

4. Methodology

4.1 Problem Framing

The problem of audio-to-gesture translation is an example of multi-modal machine learning. The term "multimodal learning" refers to a sort of machine learning in which the model is trained to comprehend and process various types of input data, such as text, images, and audio. These various data kinds reflect the various modalities, or ways in which it is perceived. In general terms, a modality refers to the way in which something happens or is experienced. More specifically, it is a problem of cross-modal translation. In our case the two modalities involved are audio and video and the translation is from speech(audio) to pose(video). The speaker video containing gesture and corresponding speech of the speaker is available to us. To construct a machine learning model capable of translating the audio features to a sequence of gestures in accordance with the speech is the problem our project aims to deal with.

4.2 Data Acquisition

We have used the dataset used in [3]. It is a large 144-hour video dataset specifically tailored to studying speech and gesture of individual speakers in a data-driven fashion. The dataset contains in-the-wild videos of 10 gesturing speakers that were originally recorded for television shows and university lectures. It has several hours of video per speaker, so that each one of the speakers' gesture style can be modeled individually. The speakers are chosen from a wide range of topics and gesturing styles. The dataset contains 5 talk show hosts, 3 lecturers and 2 televangelists.

4.3 Data Understanding

The dataset consists of video intervals for ten speakers: 'jon', 'almaram', 'angelica', 'rock', 'ellen', 'chemistry', 'shelly', 'seth', 'conan', and 'oliver'.
We have two csv files available to us: **'videos_links.csv'** and **'intervals_df.csv'**.

'videos_links.csv' file contains all the video links for all the speakers.
Shape: 2710 rows $\times$ 3 columns

Table 4.1: **videos_links.csv**

Column Name	Description	Type	Unique
'speaker'	Name of the Speaker	String	No
'video_fn'	Video's full name	String	Yes
'video_link'	Link to the video	String(URL)	Yes

'intervals_df.csv' file contains all the intervals in the videos included in 'videos_links.csv' which have clean gesture and speech.
Shape: 59482 rows × 6 columns

Table 4.2: **intervals_df.csv**

Column Name	Description	Type	Unique
'speaker'	Name of the Speaker	String	No
'video_fn'	Video's full name	String	Yes
'dataset'	'train','test', 'dev'	String Enumeration	No
'start_time'	Start time of the interval	Time	No
'end_time'	End time of the interval	Time	No

Video links for some of the videos were missing and some were dead links. Such records were discarded.

4.4 Data Selection

The aim is to learn a gesture style of a specific speaker. So, we need to select a speaker whose style we intend to generate. We decided to carry this out for Conan O'Brien, an American television host, comedian, writer, and producer. Conan O'Brien is most known for his nearly 28 years as the host of late-night talk shows, including Late Night with Conan O'Brien (1993-2009), The Tonight Show with Conan O'Brien (2009–2010), and Conan (2010–2021) on the cable network TBS. He is an articulate speaker with immaculate gesture. The tables corresponding to the two csv files were natural joined on the 'video_fn' and selection was performed for the rows which had 'speaker' as 'conan'. The query is captured by the relational-algebra expression below:

$$\Pi_{video\_fn,video\_link,interval\_id,dataset,start\_time,end\_time}(\sigma_{speaker='conan'}(videos\_links \bowtie_{videos\_links.video\_fn=interval\_df.video\_fn} interval\_df)) \quad (4.1)$$

This provides all the videos only of Conan and the clean intervals in a single table. It is saved to a separate csv file of its own and we need to concern ourselves only to this file which is related to Conan.

There are total of 3959 intervals from 326 videos of Conan. The sum of duration of all intervals is over 15 hours. With a sample rate of 20 Hz, this corresponds to over 1 million frames.

Of the 326 videos, we decided to train only on 100 of the videos. The total number of intervals resulted from 100 videos is 1008. The total duration of the intervals in the 100 videos is over 4 hours with nearly 3 hundred thousand frames when sampled at the sampling frequency of 20 Hz.

4.5 Feature Extraction

Feature extraction refers to the process of transforming raw data into numerical features that can be processed while preserving the information in the original data set. The input features are the MFCCs extracted from the audio. The output features are the sequence of poses extracted from the video. MFCCs and poses are extracted for all the intervals in the selected videos. This is done for a fixed regular duration to match a sample frequency of 20 Hz. This sample frequency is a design choice.

4.5.1 Pose Estimation

Before performing the pose estimation, it is necessary to download all the videos in the dataset. We used the Python API for OpenPose to estimate pose for the frames in the intervals in dataset, sampled at 20 Hz rate. The body keypoints, left hand key points, and right hand keypoints are extracted while the face keypoints are not considered. Body keypoints are given as a sequence of 25 points and each of left hand keypoints and right hand keypoints are given as a sequence of 21 keypoints. The indexing of keypoints is based on a specific format.

Below is a sample body pose in some random frame estimated by the OpenPose model. We have not shown all the predicted keypoints. Few of them have not been plotted intentionally.

Figure 4.1: Input image to the Openpose Model.

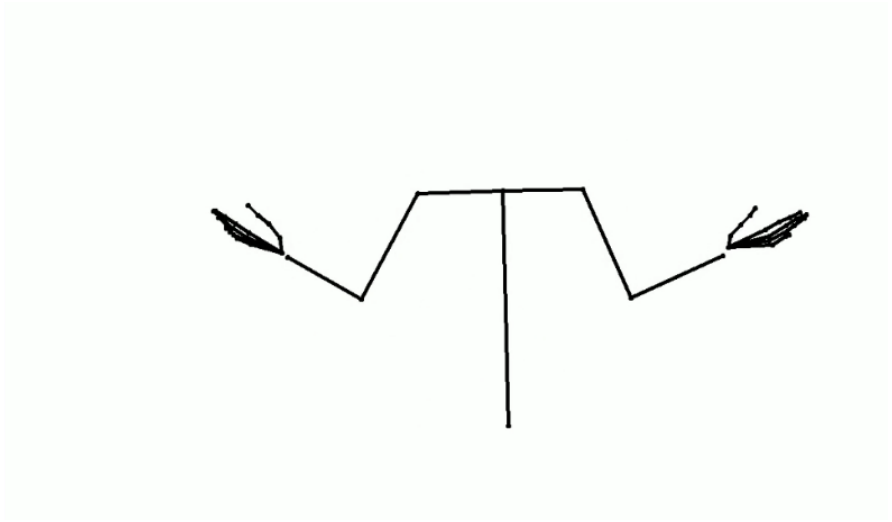


Figure 4.2: Visualization of extracted keypoints from Openpose Model.

4.5.2 MFCC extraction

Before calculating the MFCC features, we extracted the audio from the video with the sampling rate of 44100 Hz. To create a one-to-one correspondence between MFCCs and pose, we chose the hop length for calculation as 441 and the default windows length of 512. Then every five other MFCCs were averaged to match to one pose. We implemented this task using Python's Librosa library.

Below is a visualization MFCCs of some random sample in the dataset.

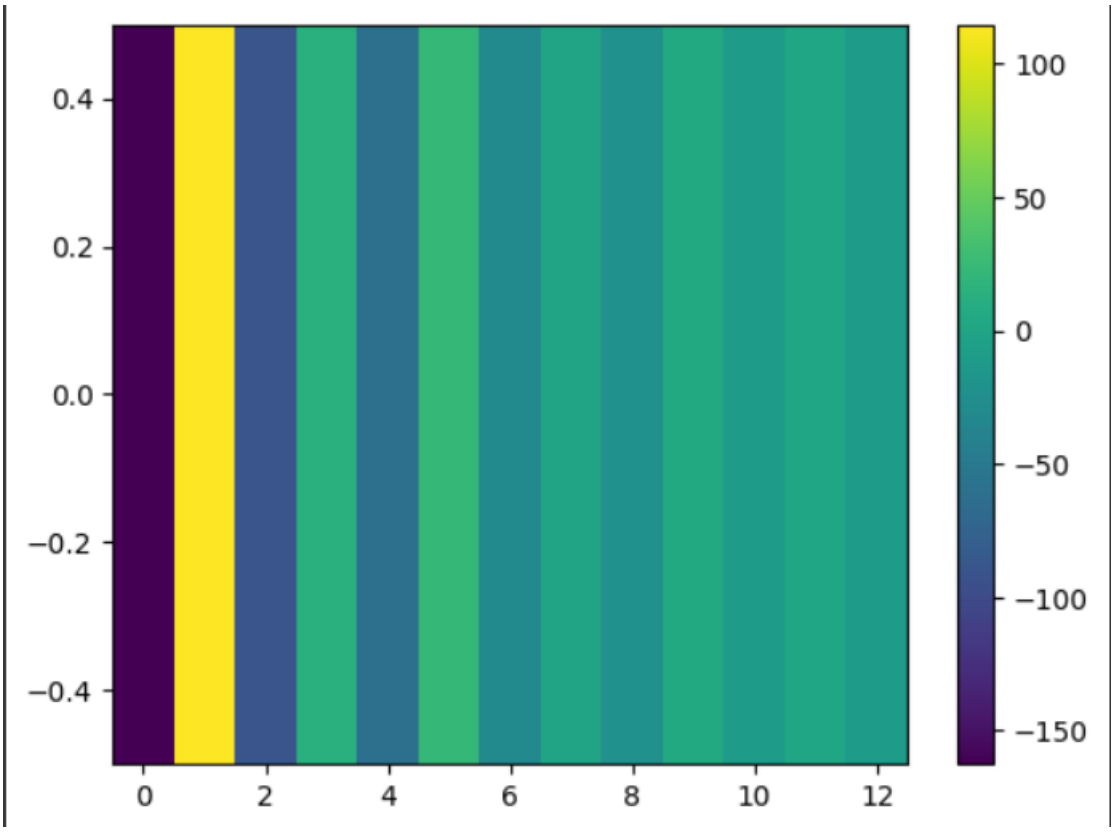


Figure 4.3: Visualization of MFCC features of a random sample without temporal context.

However, to make generated gestures more plausible, temporal context i.e. using 20 samples before the current sample and 20 samples after the current sample, was found to be more reasonable. As a result, instead of using MFCCs features of current sample only, we used MFCCs features based on temporal context. Therefore, MFCC feature format of (41,13) was used for every training frame. Below is the visualization of MFCCs features with temporal context.

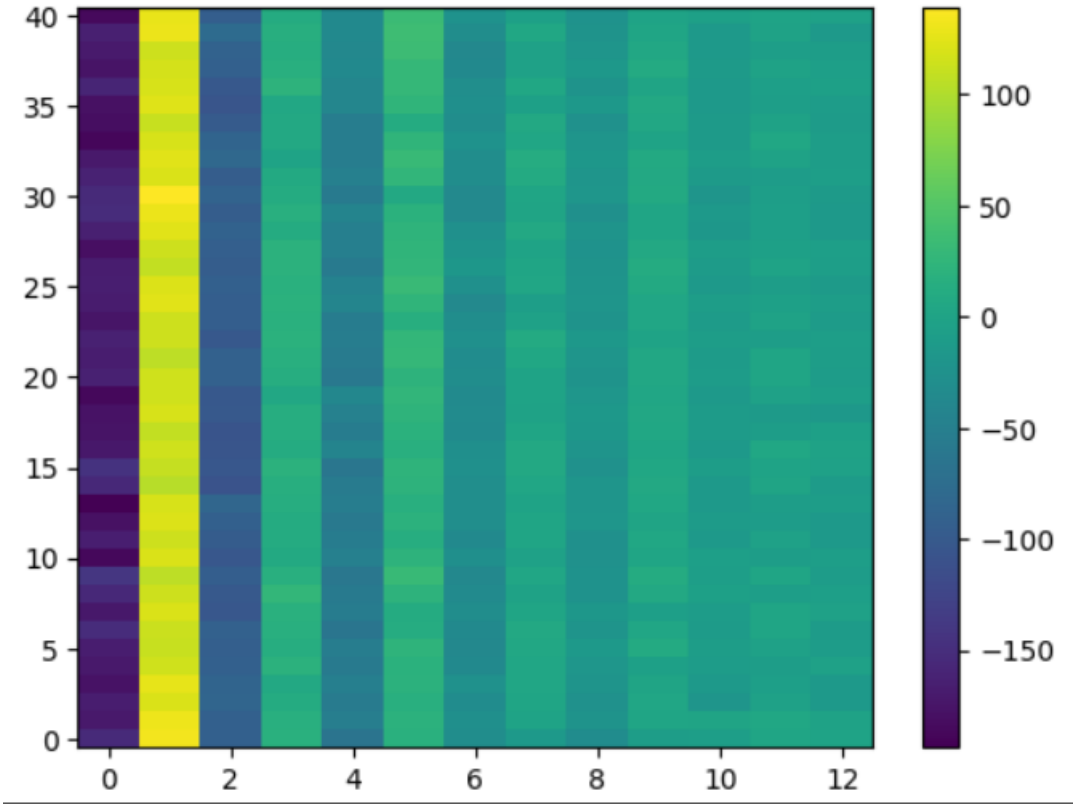


Figure 4.4: Visualization of MFCC features with temporal context.

5. System Design

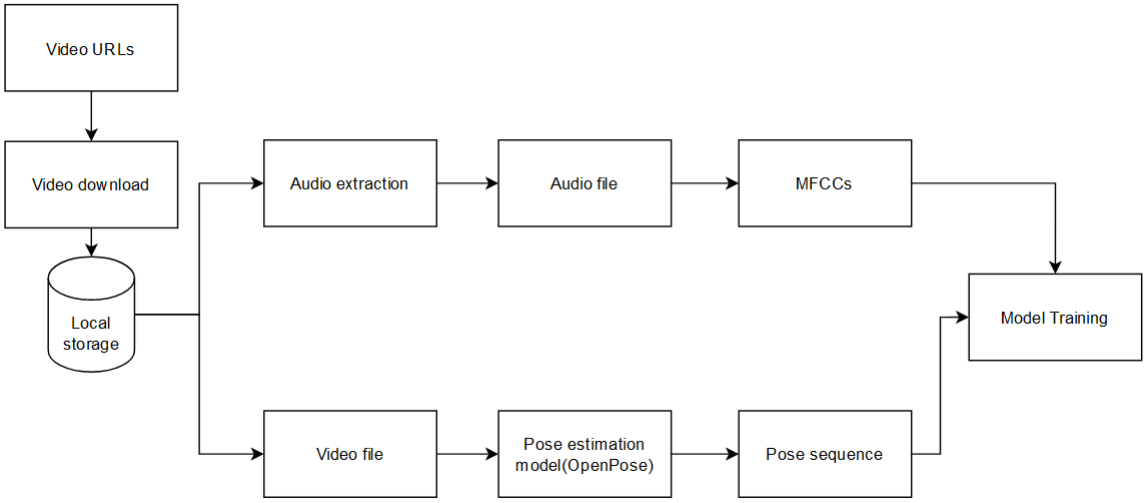


Figure 5.1: System Design.

The design for our system is shown in the block diagram above. We have video URLs available to us to begin with. The videos are downloaded to the local machine using tools like youtube-dl. In the next stage, audio and video content of the file are separated for carrying out separate processing on them. The extracted audio is processed to calculate the corresponding Mel Frequency Cepstral Coefficients (MFCCs). This is done using a Python library called Librosa. A pose estimation model is employed to estimate the pose sequence in the videos. Specifically, the pose estimation model used in the project is OpenPose. For training our models, audio MFCCs are input signals and their corresponding pose sequences are the ground truth signals.

6. Implementation

6.1 BaseLine Model

For the baseline model, we can have a model that always predicts a median pose regardless of the input audio. This is feasible if the speaker spends most of their time in a rest position. Besides, we can also have a model that predicts pose randomly. This method however is not quite plausible. We decided to go with a neural network that predicts pose without the temporal context in the input. The pose of some specific time is predicted using only the MFCCs at that specific time.

6.2 Tested Model

We carried out the implementation on a Generative Adversarial Network (GAN). We tried different configurations of GAN. However, we did not manage to lower the loss. So, we decided to try other alternatives. Also, we tried a neural network without the use of temporal context in the input data. This model also did not seem to perform well. So, we discarded this model too.

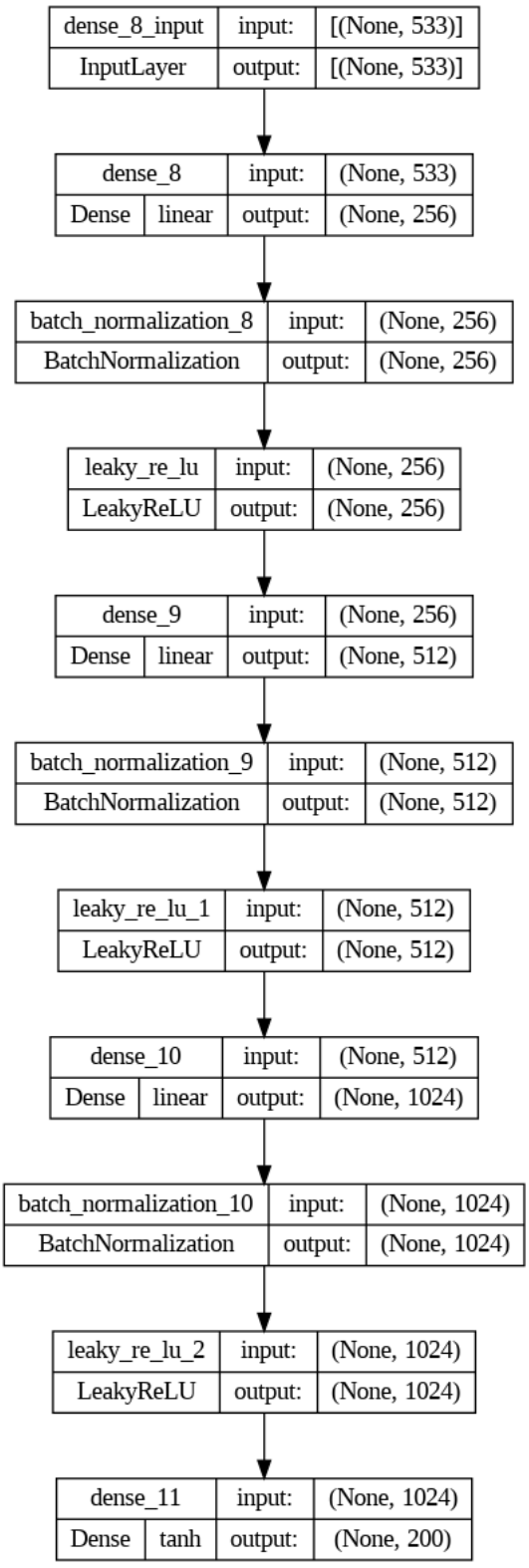


Figure 6.1: Generator architecture.

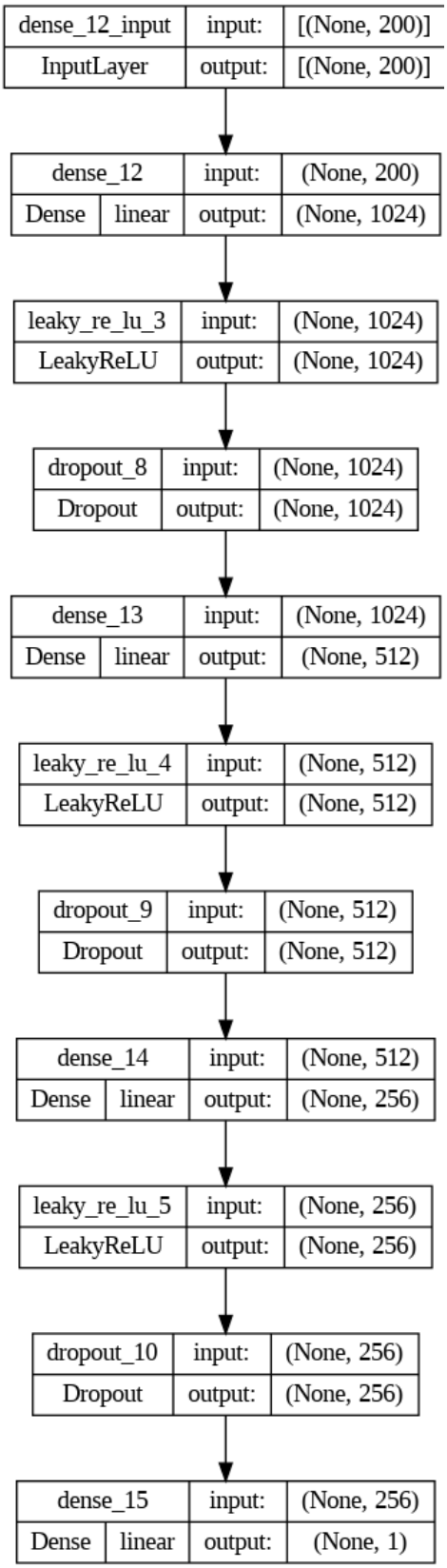


Figure 6.2: Discriminator architecture.

6.3 Implemented Model

We experimented on different models like simple ANN, GAN and RNN with GRU as a recurrent unit. Of all, we found RNN to be the best performing for our dataset. The architecture of the model is summarized in the table below:

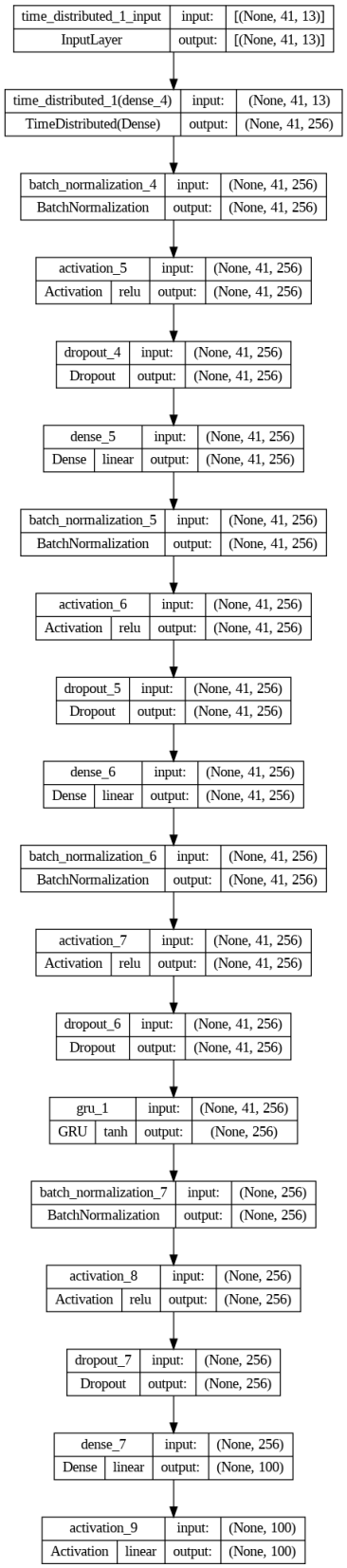


Figure 6.3: Model Architecture.

7. Model Performance

We carried out experiments with NN, GAN and RNN. Based on the model performances, we decided to choose RNN over NN and GAN. The performance evaluation of each of the models is described below:

7.1 Performance of GAN

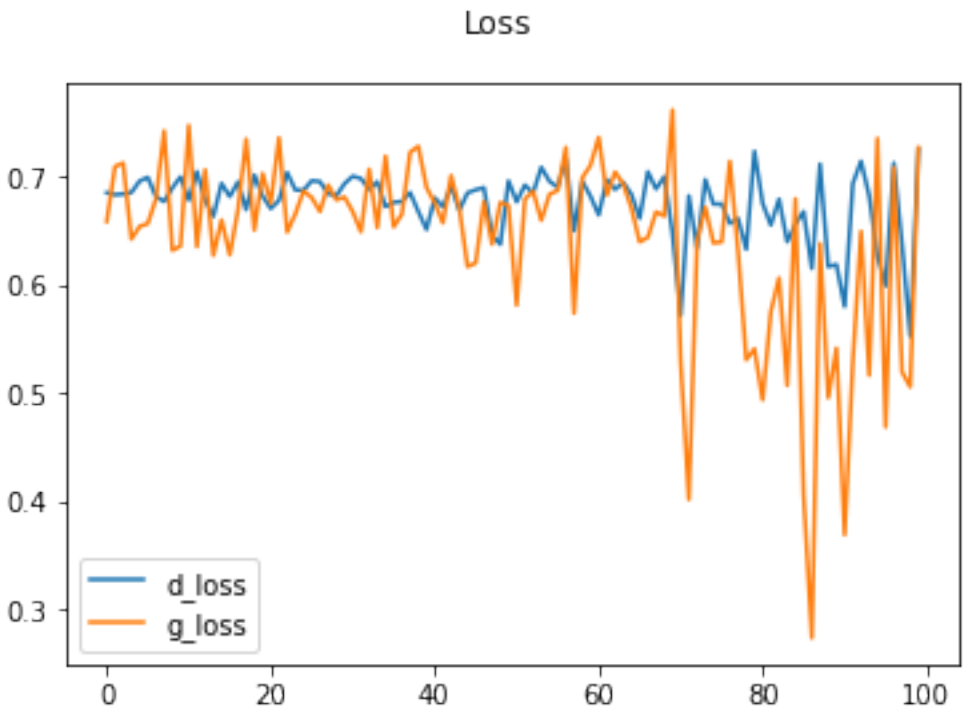


Figure 7.1: Loss curve of GAN.

The generator loss and discriminator loss values for the GAN did not seem to decrease over several epochs. Due to this reason, the loss curves for GAN is not lowering towards the right as shown above.

7.2 Performance of NN

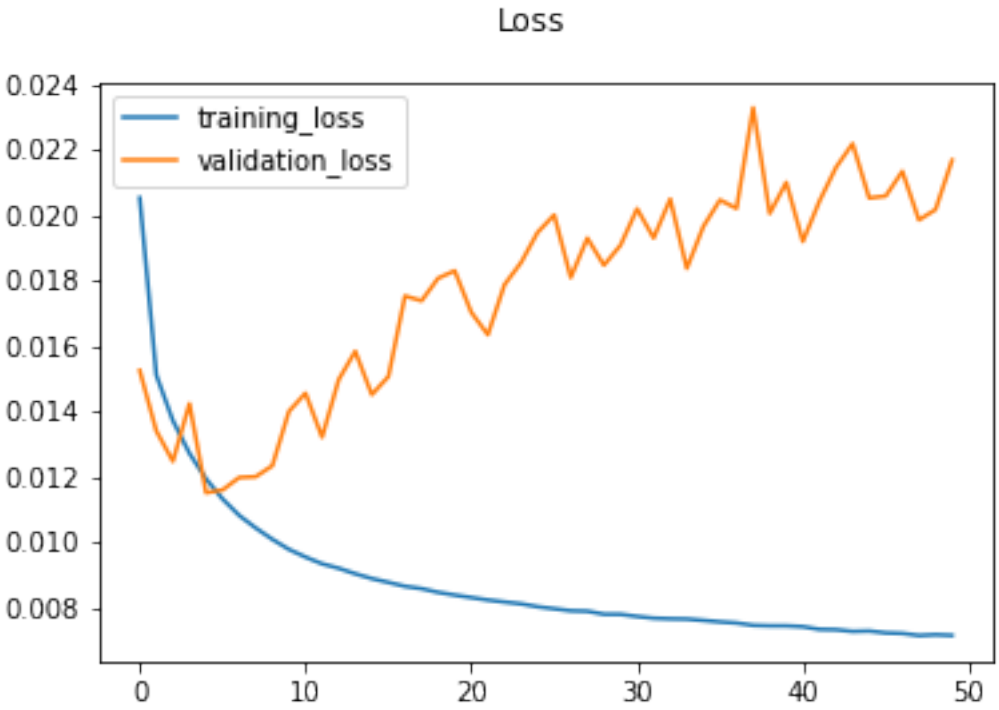


Figure 7.2: Loss curve of NN.

Although the training loss decreased sharply when training the baseline model, the model was unable to generalize to the validation dataset. The model was predicting the mean position for entire input samples.

7.3 Performance of RNN:

Baseline model with further modification was used as our implemented model. We shuffled the dataset provided to the model which significantly improved the results. We also replaced the middle time distributed dense layer with simple dense layer. Also, the learning rate was set to 0.0003, epochs to 50 and batch size to 512. In this model, we used a temporal size of 41, with 20 MFCCs previous of current time, one MFCCs at current time and 20 MFCCs from the future. This is purely a design choice and can be adjusted to be larger or smaller. This is done because the pose at some specific time depends not only on speech at present but also on the utterance in the past as well as what the speaker is about to say. Below is the loss curve for the model when training for 50 epochs.

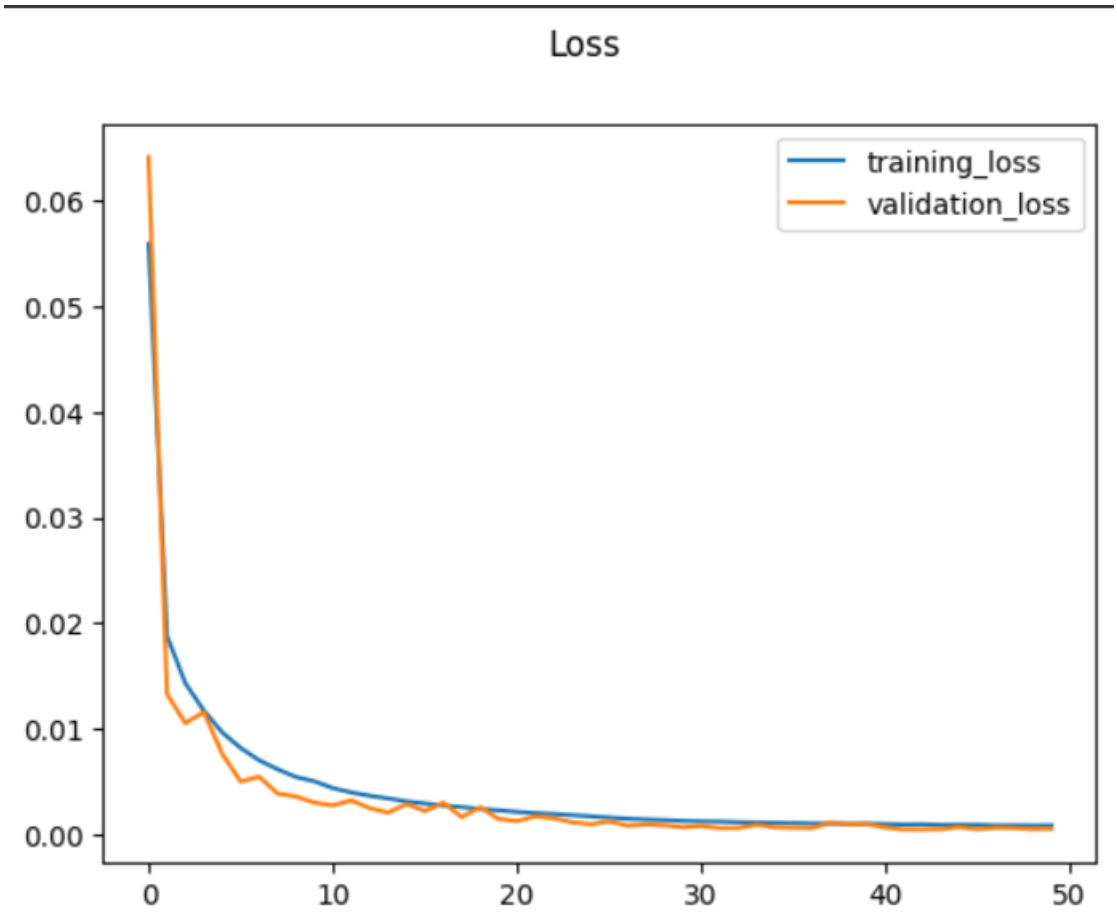


Figure 7.3: Loss curve of RNN.

Both the training loss and validation loss decrease significantly as the epoch progresses as shown in the loss curve. This is the model we used for making predictions.

8. Results

The final trained model is used to make the predictions of pose from the input audio. While qualitative analysis of the predicted gesture is subjective, the performance metric on a test data was convincing. Also, the predicted poses when put together to create a motion, it was plausible.

From the uploaded audio, first the MFCCs are extracted and the features are fed as input to the trained network which generates a sequence of keypoints. The keypoints are plotted in sequential order to match a certain frame rate.

8.1 Baseline Output

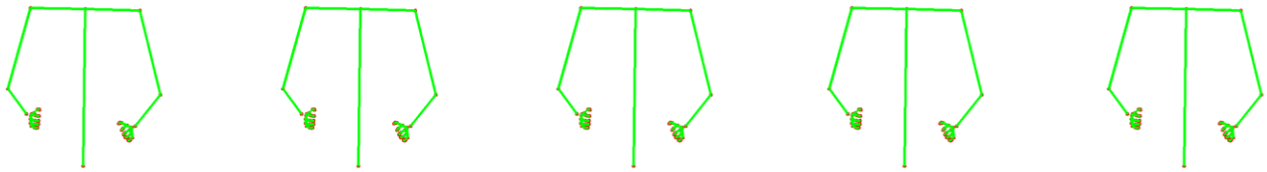


Figure 8.1: Output of baseline model.

Baseline model was converging to mean position of the training keypoints. So the model was generating same keypoints result for every input sample.

8.2 GAN Output

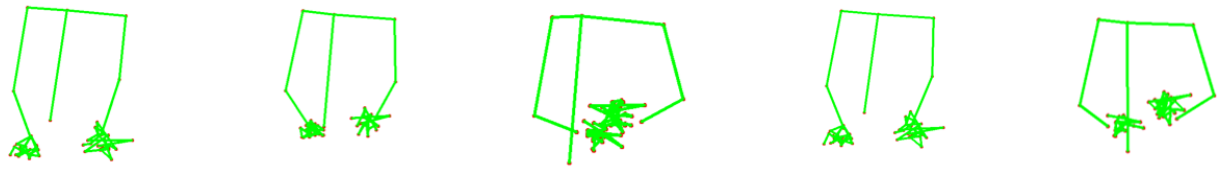
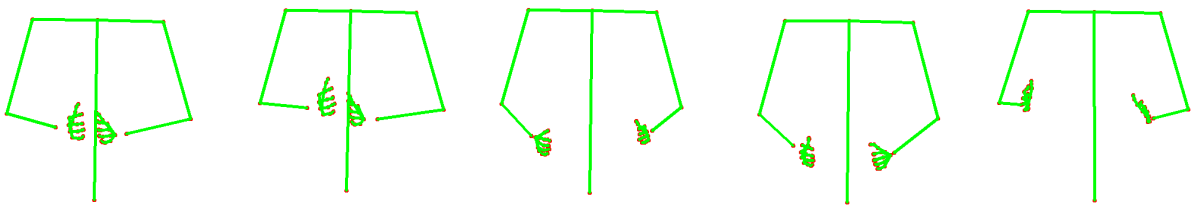


Figure 8.2: Output of GAN model.

Gan model was not able to understand hand structure which resulted in unclear hand keypoints.

8.3 RNN Output

The below figure is a sequence of predicted poses for 5 frames. The final deliverable allows user to upload a clip of speech. From the uploaded clip, a gesture sequence is predicted. The gesture sequence and the audio clip are played simultaneously. This video clip is our project's result.



9. Limitation

Audio can provide some information about the speaker's emotional state and tone of voice, but it is not always accurate. Limited expressiveness, noise and interference, and the quality of the audio used for gesture synthesis can all have an impact on the accuracy and effectiveness of the process. Cultural and linguistic differences can also make it difficult to accurately synthesize gestures using audio, as what may be considered a common gesture in one culture may not be used or recognized in another culture.

10. Conclusion

The project successfully implemented a gesture synthesis system that generates realistic and diverse gestures based on input parameters such as mfcc. The system was evaluated using various metrics such as naturalness, expressiveness, and coherence, and showed promising results compared to existing methods.

The project identified some limitations and challenges of the current approach, such as the need for more training data or more sophisticated modeling techniques to handle complex gestures. The major hinderance we faced during the course of this project was to extract keypoints from training videos using OpenPose.

Future work could include expanding the scope of the system to generate gestures in different domains or for different applications, or integrating the system with other modalities such as speech or facial expressions to create more immersive and natural interactions.

11. Future Possibilities

There is still much room for future extensions and improvements to our project. Here are some potential areas for future extension of gesture synthesis:

- Improved Naturalness:
  One of the significant limitations of gesture synthesis is the lack of naturalness in the generated gestures. Researchers could explore ways to improve the naturalness of generated gestures by incorporating more sophisticated algorithms and machine learning techniques. For instance, developing algorithms that can better capture subtle differences in body language, facial expressions, and other nonverbal cues could enhance the naturalness of the generated movements.

- Better Data Collection:
  Gesture synthesis algorithms require significant amounts of data to learn and generate realistic movements. However, collecting and curating this data can be time-consuming and challenging. Researchers could explore ways to improve data collection methods, such as developing new sensors and cameras that can capture more precise movement data. They could also work on creating larger and more diverse datasets to improve the accuracy and range of the generated movements.

- Cross-Cultural Gestures:
  Gestures can vary significantly across cultures, and gesture synthesis algorithms may not be able to account for these differences. Researchers could explore ways to develop cross-cultural gesture synthesis algorithms that can generate movements that are appropriate and relevant across different cultures. This could include creating gesture synthesis algorithms that can learn and adapt to different cultural contexts and nuances.

- Multimodal Gestures:
  Gestures are not just limited to body movements; they can also include facial expressions, vocalizations, and other nonverbal cues. Researchers could explore ways to develop multimodal gesture synthesis algorithms that can generate movements across different modalities. This could enhance the naturalness of the generated movements and make them more human-like.

- Personalized Gestures:

  Gestures are highly personalized and can vary significantly between individuals. Researchers could explore ways to develop personalized gesture synthesis algorithms that can generate movements that are specific to an individual's body shape, size, and movement patterns. This could lead to more natural and authentic movements that are unique to each individual.

12. Timeline

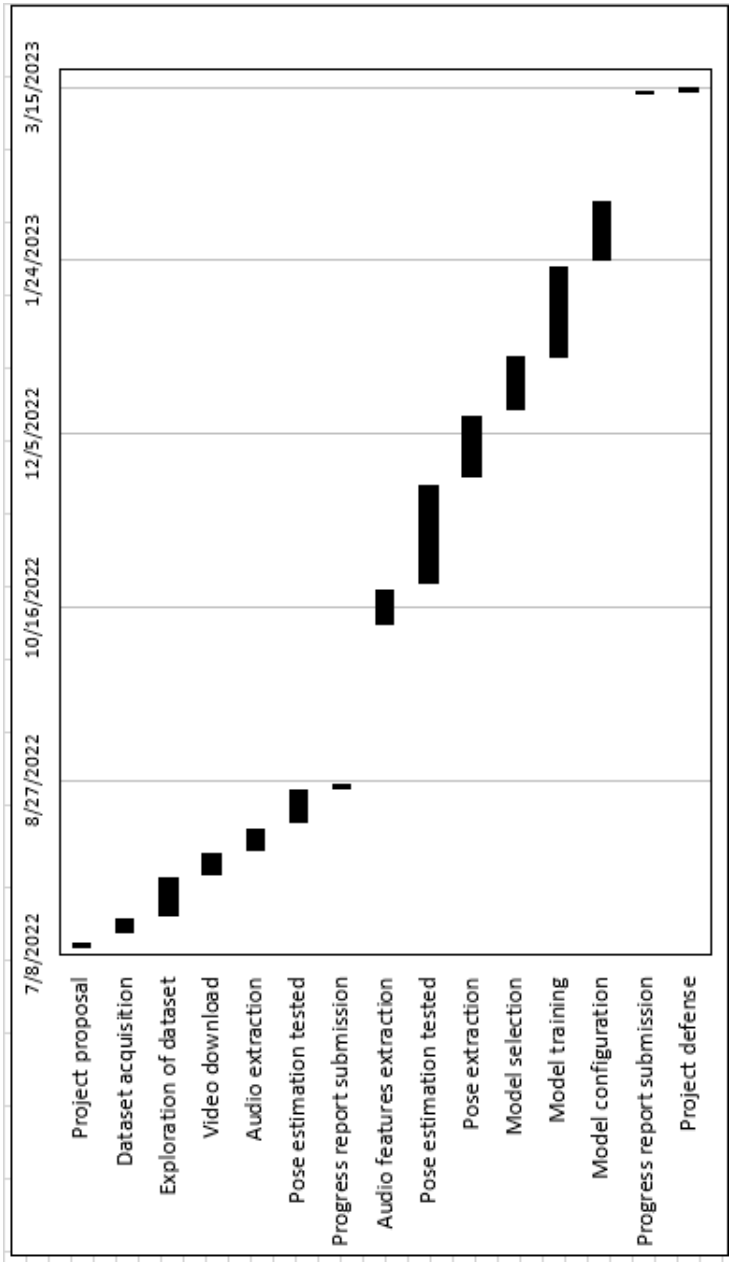


Figure 12.1: Gantt Chart.

References

[1] Susan D Fischer. Hand and mind: What gestures reveal about thought. by david mcneill. chicago: University of chicago press, 1992. pp. xi, 416. 1994.

[2] Adam Kendon. Gesture: Visible action as utterance. 2004.

[3] Shiry Ginosar, Amir Bar, Gefen Kohavi, Caroline Chan, Andrew Owens, and Jitendra Malik. Learning individual styles of conversational gesture. pages 3497–3506, 2019.

[4] Chaitanya Ahuja, Dong Won Lee, Ryo Ishii, and Louis-Philippe Morency. No gestures left behind: Learning relationships between spoken language and freeform gestures. pages 1884–1895, 2020.

[5] George Papamakarios, Eric T Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing flows for probabilistic modeling and inference. *J. Mach. Learn. Res.*, 22(57):1–64, 2021.

[6] Simon Alexanderson, Gustav Eje Henter, Taras Kucherenko, and Jonas Beskow. Style-controllable speech-driven gesture synthesis using normalising flows. 39(2):487–496, 2020.

[7] Shenhan Qian, Zhi Tu, Yihao Zhi, Wen Liu, and Shenghua Gao. Speech drives templates: Co-speech gesture synthesis with learned templates. pages 11077–11086, 2021.

[8] Bowen Wu, Chaoran Liu, Carlos T Ishi, and Hiroshi Ishiguro. Probabilistic human-like gesture synthesis from speech using gru-based wgan. pages 194–201, 2021.

[9] Tadas Baltrušaitis, Chaitanya Ahuja, and Louis-Philippe Morency. Multimodal machine learning: A survey and taxonomy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(2):423–443, 2019.