|  |  |  |
| --- | --- | --- |
|  |  |  |

**Egypt IOT and AI Challenge 2020**

**Graduation Projects Track**

**Detailed Project Progress Report GP (20-57)**

**Arabic Sign Language (ARSL) Interpreter**

**------------------------------------------------------------------------------------------------------------------------------------**

**Submitted By**

|  |  |
| --- | --- |
| **Mahmoud Aboud Nada**  **MennatuAllah Ahmed El-Zahaby**  **Hesham Abdullah Al-Kholey** | **Asmaa Hasan Dardarah**  **Muhammad Ali Shokair**  **Omar Ibraheem Shaheen** |

**Supervisor**

**Professor/ Wessam Fekrey**

**DEPARTMENT OF ELECTRICAL ENGINEERING**

**KAFRELSHEIKH UNIVERSITY**

2020

# 1. Project Presentation and Background

## 1.1. Problem Statement and Motivation

According to the world health organization (WHO), over 5% of the world’s population suffer from deafness or hearing impairment. In Egypt alone, there are around 7.5 million people who are suffering from this problem.

Inability to speak or hear is considered to be a true disability. People with this disability use different modes to communicate with others, there are number of methods available for their communication one such common method of communication is sign language. Sign language allows people to communicate with human body language; each word has a set of human actions representing a particular expression.

Deaf and mute people can often feel isolated or lonely. This is because they can have difficulty interacting with people on a day-to-day basis, so they feel disconnected from friends, family, their community and the world around them. They want to express themselves, be independent, live life to the fullest and do all the things like people without disabilities. We aim to help them leave loneliness behind; Just as importantly, give them the confidence to reconnect with their family, friends and community, and embrace the life they want to lead. That’s why our motive is to convert the Arabic sign language (ARSL) to spoken voice through human gesture understanding and motion capture models, so they can express what they want to say and everyone would understand.

## 1.2. Goals, Objectives, and Outcomes

## Our main goal is to ease the lives of people suffering from deafness and muteness. We aim at creating a device that can be worn by the deaf and mute people that can capture their Arabic sign language and interpret it to spoken language through microphone so other people who don’t know sign language can understand what they want to say. The users of the device will be able to carry it with them everywhere and adjust its posture as they use it. We aim at creating a device that looks somewhat like the following pictures.

## 

Figure 1.1: sample views of our final product

## The main purpose of our project and product is to ease the daily life communications for the deaf and mute community members. We are offering a way where they will be able to perform all communication activities without help from other people, they will no longer be alone in the society and they will have better work opportunities and experiences.

Our project has many strengths such as:

* it offers Arabic Sign Language interpretation. ARSL has no interpretation systems that are fully functional till now.
* It’s an easy to use device that doesn’t require the user to wear any gadgets on their face, or their hands, they only need to carry around our small device.
* It offers observation of surroundings and alarm notification in case of any dangers.
* It can be turned into a mobile application as an alternative to carrying the device.

## 2. Methodology, Implementation and Results:

## To build a system that can translate full Arabic sign language sentences and turn it from signs to spoken words and turn this idea to a product that can be used by everyone in the deaf and mute community, we had to go through a lot of different phases and follow many steps.

**Step #1**

First, we had to gather our own customed dataset that consists of a set of words that we believe are most common in daily life according to sign language instructors. we started with 28 words from 4 different signers each word is repeated 10 times so this makes our initial dataset with 1120 videos.

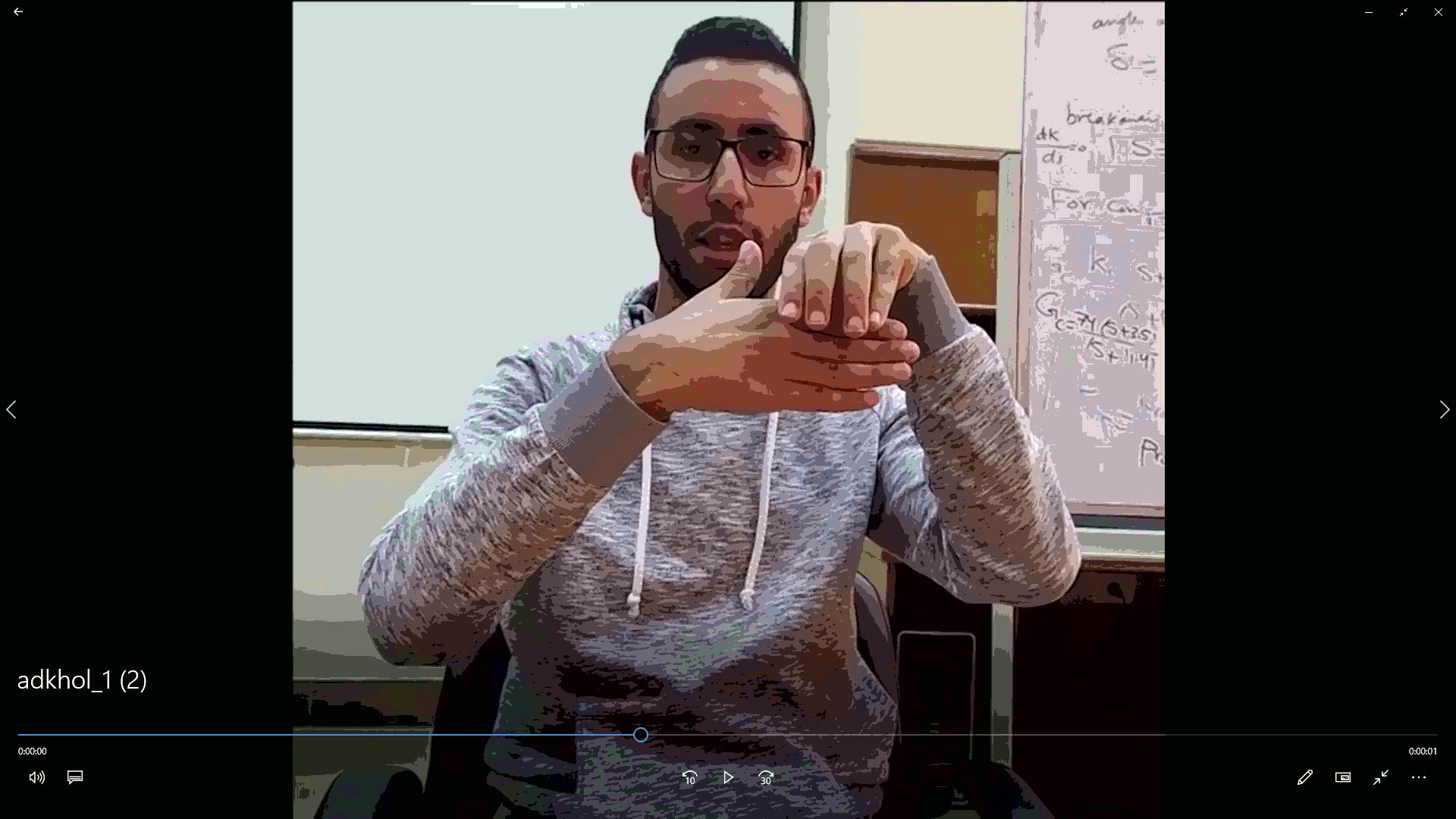


Figure 2.1: Snapshots from the ARSL video dataset

As we were collecting the dataset from local sign language specialists, we started to prepare the code that deals with any video dataset, we used an experimental dataset called UCF101 that is used in action recognition it has 101 classes within each class videos belongs to this class.

**Model #1**

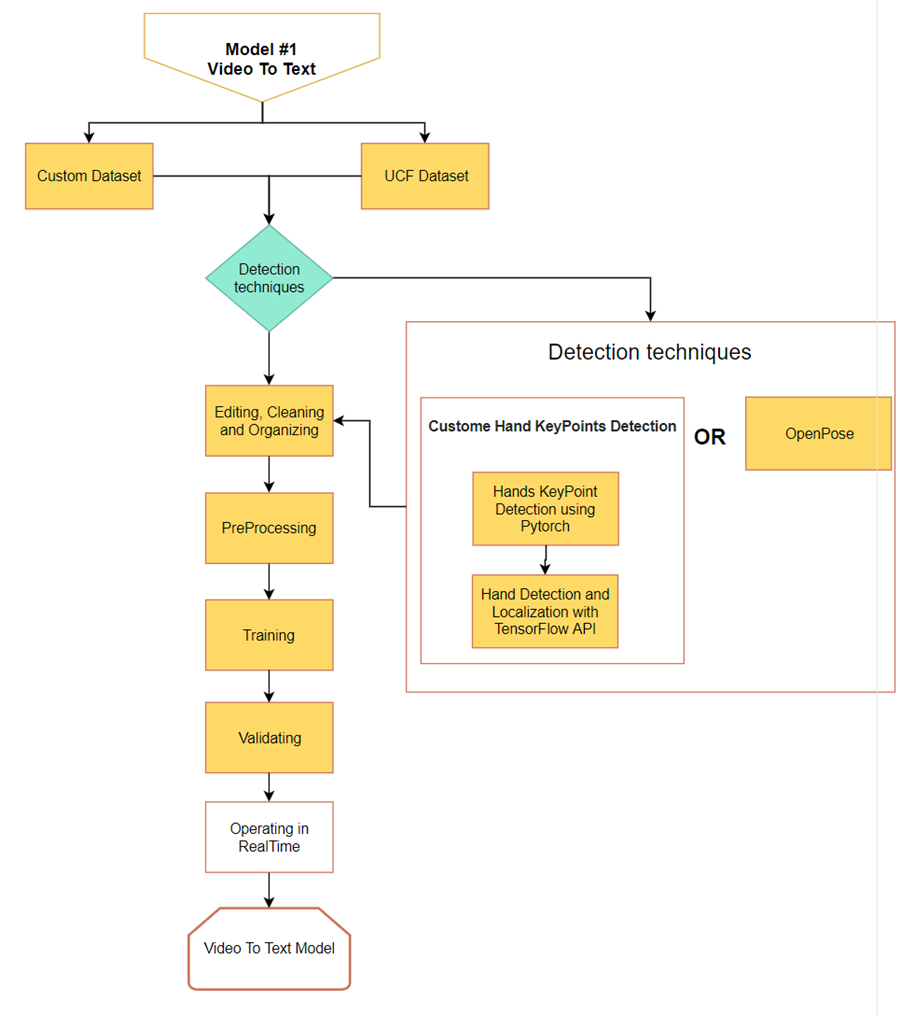
****

Figure 2.2: Model 1 graph

First, we started working on the video to text conversion without going into the detection techniques and methods.

With the help of UCF dataset, we managed to build the code that preprocess the videos (organize them and extract the frames from each video using ffmpeg tool), we also built an initial small CNN + LSTM model that we used in training.

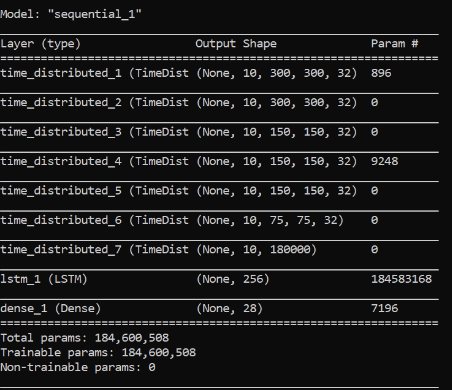
Figure 2.3: The initial custom model for training

Figure 2.4: The custom model architecture

To try this model we took a segment that belongs to one signer and started with it to see whether our concept is capable of learning the patterns or not; each video of the dataset has around 30 frames we decided to pick 10 frames from each video as a start also resized all frames to be (300 x 300) this makes our input shape = 10 x 300 x 300 x 3, our model performed naively well and had an accuracy of 100% at training and validation, when we tried this model on a video that was preprocessed and had its frames extracted we got a false prediction of the sign, we believe that 10 frames from each video was not a good choice so we are changed it to be near the minimum number of frames at the videos in the dataset which is 30 frames and it did perform really well.

The next step is to train the model with 4 signers’ videos, enhance and tune a model to get the best results.

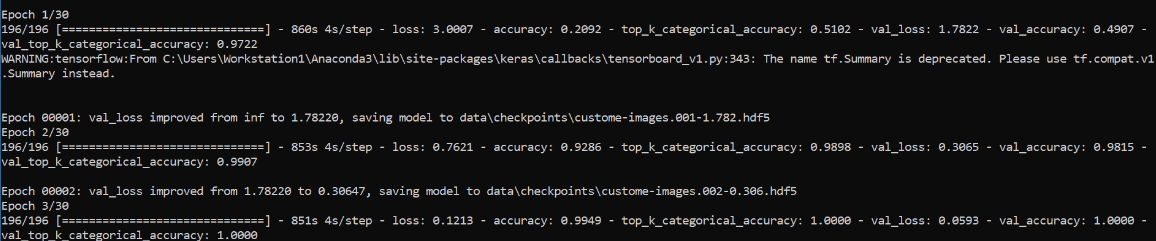


Figure 2.5: Snapshot from the training process

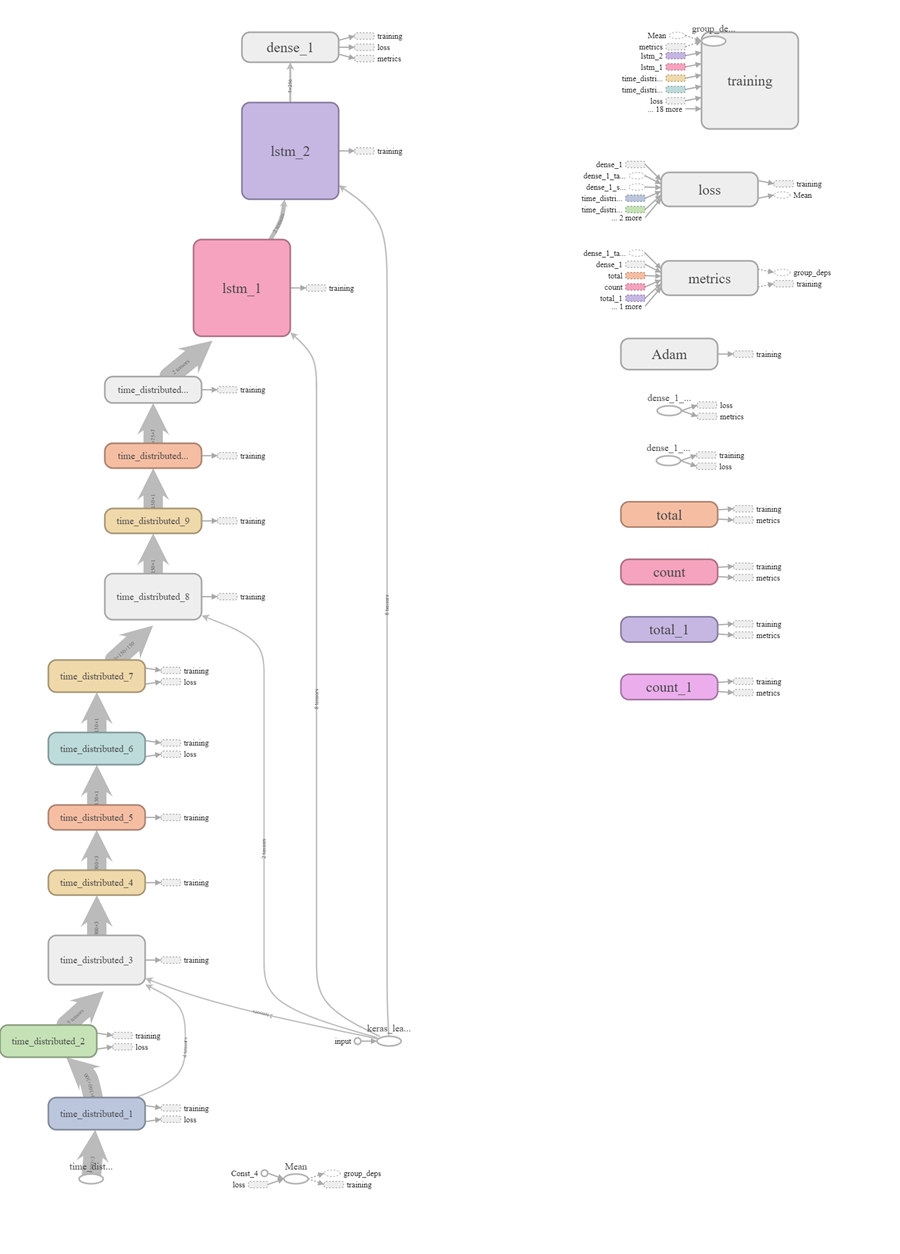
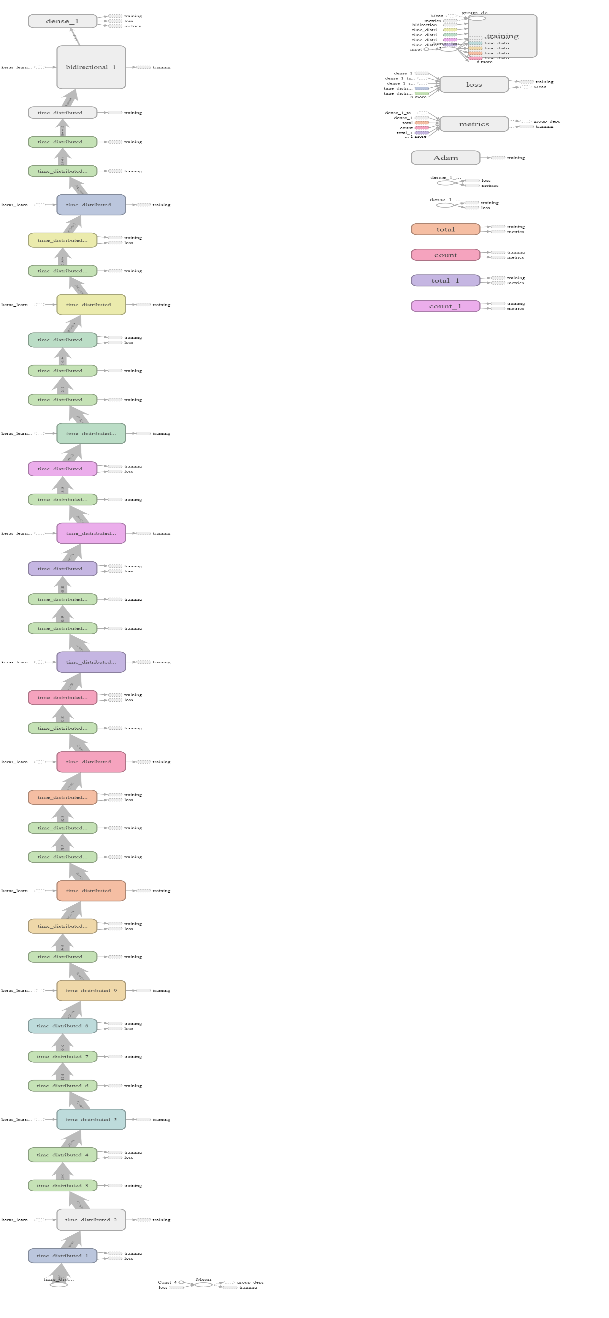
We added multiple Conv2D layers to extract better and more features that the LSTM can learn from and added another LSTM that takes the sequence from its prior LSTM and produces a vector to the output Dense layer.

Figure 2.6: The modified custom model architecture graph

The Previous Model Did not Perform significantly better that than the first one so we used a similar architecture introduced in an Action Recognition Paper called [Very Deep Convolutional Networks for Large-Scale Image Recognition](https://arxiv.org/abs/1409.1556)**,** and used a Bidirectional LSTM at the end as advised by the [Deep ASL Paper](https://arxiv.org/abs/1802.07584) and this is our Final Model Architecture.

****Figure 2.7: Final model Architecture

Training this model took very large time, but was robust and has better results than the priors so we saved its weights to use in the next step.

The final step in this model is to build a script that takes a video as frames and pass every sequence of frames to our model to be interpreted.

The following segment shows part of the training process.

Figure 2.8: Training process of the enhanced model through days of fine tuning.

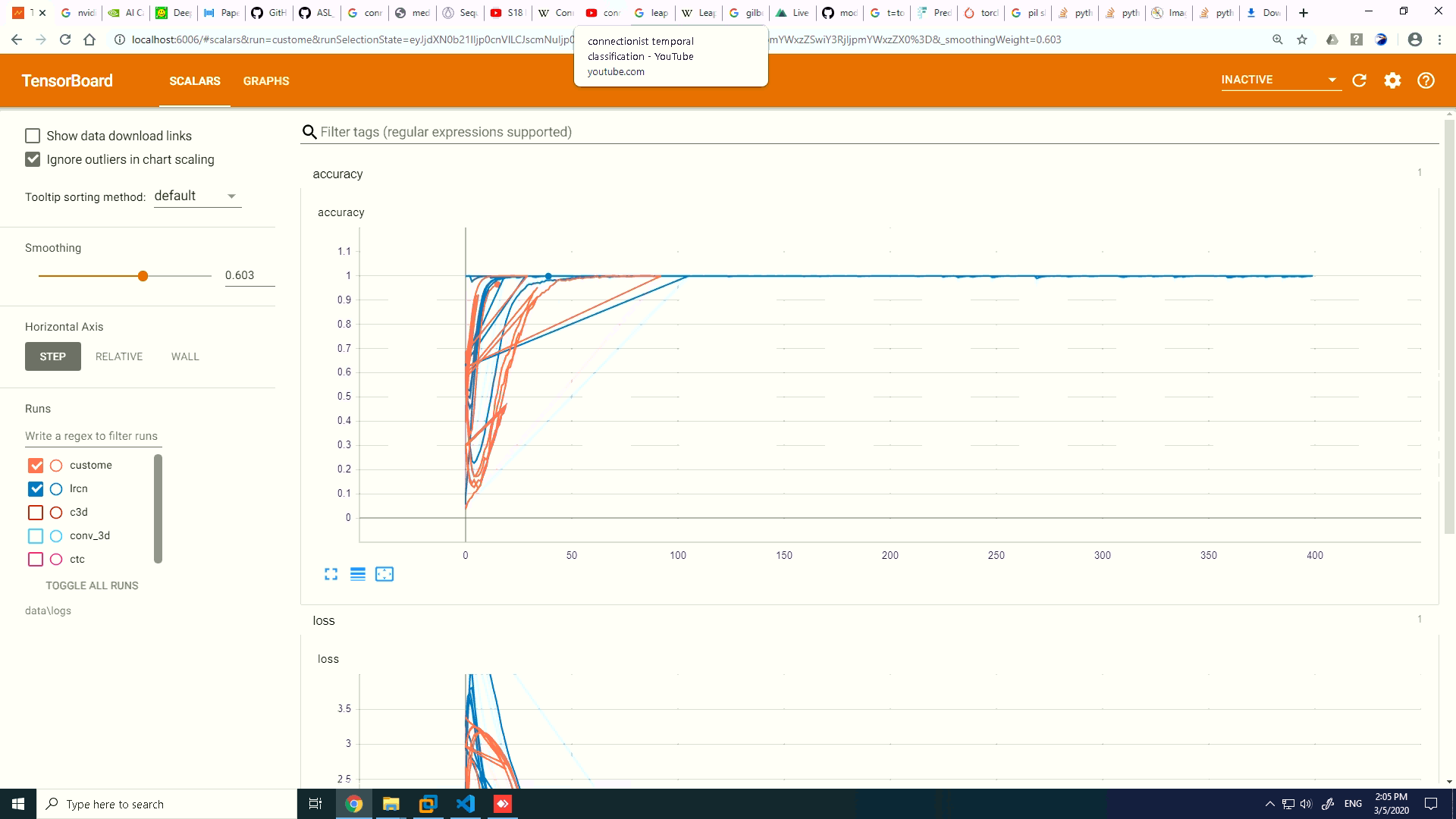
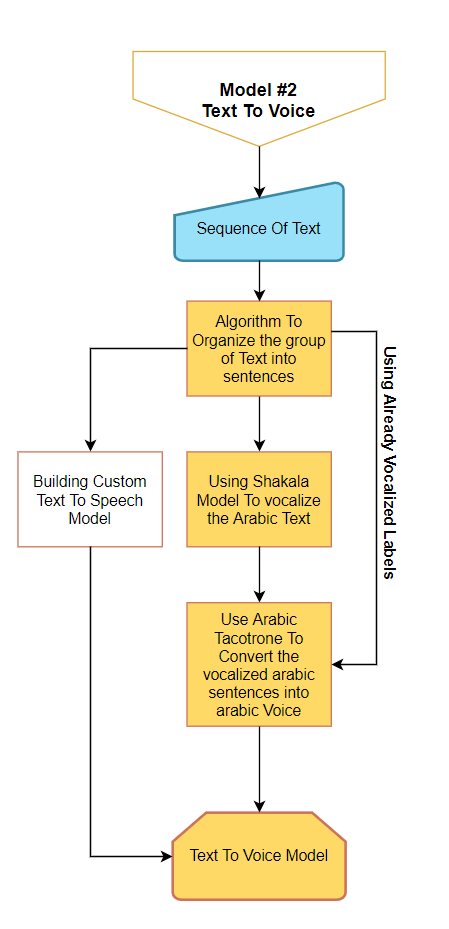


Figure 2.9: results of the different model we have tried

**Model #2**

The second model converts Arabic text to Arabic voice to do so, we used the help of two separate models:

* First, we vocalize the text using a model called Shakkala; this model takes in Arabic unvocalized text and returns the vocalized version of this text.
* Second, we used an Arabic text to voice model that is based on tacotrone model; this model takes in the vocalized text and produces generated Arabic voice as output.

We had several other theories on how to generate the voice without the need to go deep into several models, we thought of building a model that directly generates speech from the received video frames without the need for the text middle step, but we faced the problem of the dataset structure. We also thought of building a model to take the unvocalized text

Figure 2.10: Model 2 graph

as input and result in Arabic voice without the need for the vocalization method. In the end we decided to stick with our first approach as it has given us the best results for now; and leave the other approaches for our improvements phase.

**Deployment**

Deploying a Deep learning model is not an easy task, especially in our case where we have multiple models running simultaneously in addition to the fact that these models need to be running in real time as well as being portable e.g. (Phone).

**Available Deployment Approaches:**

**First:** We can save the models (freeze their weights) and use them on power computer with huge GPU power.

**Pros:**

This will make deployment a lot easier and save a lot of time as we do not need to worry about computation power needed to operate these models

**Cons:**

1. Lack of Portability
2. Inaccessibility to normal users
3. The device will not be feasible to normal users

**Second:** transform all these models to TensorFlow Lite models that has small size and can run on mobile Apps.

**Pros:**

1. Great portability advantage
2. Memory optimization
3. No need to connect to the internet to get the Service

**Cons:**

1. A lot of time needed to transform and edit these models
2. The performance will decrease due to transformation downgrading
3. Even if all the above steps completed successfully, we still do not know for sure if it will be fast enough on mobile phones
4. Not all phones would be able to run such application

**Third:** we buy a high computation server to deploy our model on, this way the model can be accessed by mobile applications or normal websites without being slow.

**Pros:**

1. Compatibility with every software that connects to internet
2. This way any lightweight app that runs on all mobile phone would be able to get the service, this way any user with a connection to internet would get the service.
3. This increases the feasibility and portability and that the core advantage

**Cons:**

1. Internet connection that handles streaming is not available everywhere like in our target case, Egypt.

After many researches and experiments we chose to go with the third deployment method, and the following steps are to be made:

* Create a website using flask that can talk directly to the model, it sends a video feed to the model and receives the interpretations back.
* Look up a server that is affordable to us in the meantime, to deploy our system on.
* Last step is making an API that takes the video feed from the Flask website and gives us back the interpretations.

**Current Results**

For our prototype, we built a simple web page using HTML and JavaScript. The web page receives a video then processes it extracting the frames and saving them into a list, after that it calls our Prediction model passing in the preprocessed frames to predict the corresponding word (label) for the given video from our labels list and return the Arabic word as a message through our page with the corresponding voice for the predicted word.

**Future Work**

Our current results aren’t perfect and we have many future plans for our project and device to be as helpful as possible to everyone in the deaf and mute community. For now, we are working towards two main aspects:

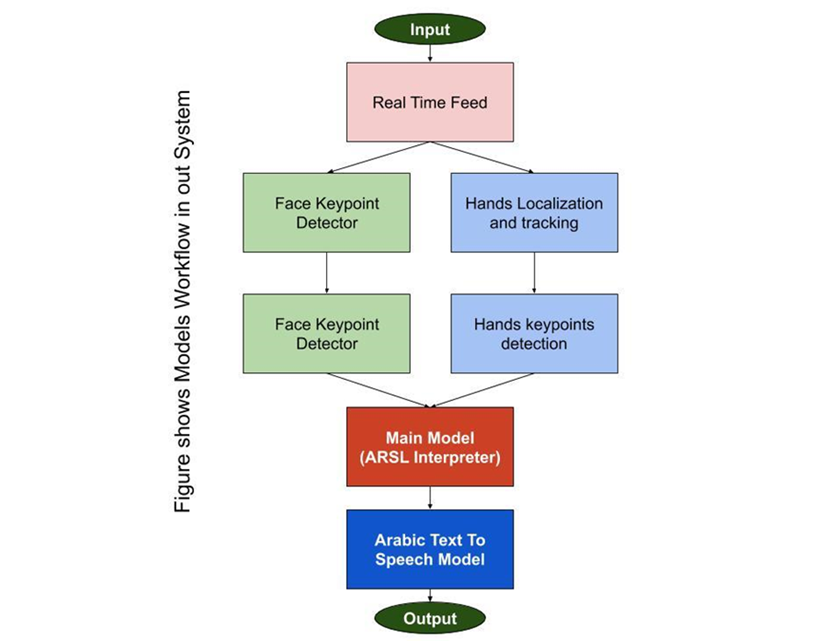
1. Build a more consistent model that receives real time signs which it can detect the key points of the signer’s hands and face, extract those key points and save them as a dataset of landmarks. The model then trains on those key points landmarks and pass straight to the prediction algorithm where it can predict the correct corresponding label to each word and generate the correct voice for that word in Arabic. It should be a fast working model that’s able to process in real time.
2. Deploy the new model on the hardware device using Raspberry Pi as our main controller with the needed gadgets, a camera, microphone, speaker, etc. the users will use this portable device whenever they need to talk to anyone that is not familiar with Arabic sign language easily and reliably.

Figure 2.11: The final project flow graph

Figure 2.11, show the desired final workflow graph that we are aiming at when we finish our work. It’s a smooth workflow and consistent operation between models, leading to accurate and fast results in real time.