Final Year Project – ASL Recognition

**Production Log 1: (Friday, 26/01/2024, 11:00 am)**

This is a real pain having to write down and summarize everything in words, but it is what it is.

I am currently running the project through a single Jupyter Notebook file and using an external hard disk to store all the extracted content and the dataset. Till now, I have been running the project on my Hp Envy Laptop but will later be using the university laptop to train and run the model.

My plan is:

* Stage 1: Create the dataset from videos.
* Stage 2: Train proposed ResNet + BiLSTM model with data.
* Stage 3: Train 1 or 2 other models. (we’ll decide them later)
* Stage 4: Evaluate results.
* Stage 5: Real Time Detection using Webcam. (have found code for this)
* Stage 6: Convert predicted Results to audio. (have found code for this)

I am currently in stage 1 of production. So, this is what I have worked on till now:

1. Decided on top 11 classes to train the model with (classes chosen based on the number of videos per class).

The number of videos per class lies in the range 9 to 14.

1. I have found a method to extract frames from each video.

Since the videos are not of equal lengths, each video will have unequal number of frames extracted. To overcome this, 60 frames will be extracted from each video. If the number of frames extracted is less than 60, then the last frame is repeated until the frame count reaches 60.

1. I have found a way to apply MediaPipe Holistic to a frame.

This detects and draws landmarks for face, hand and pose.

1. As for storing these frames, the folder hierarchy is as follows:

Main frames folder -> folders for actions/classes -> folders for each video in the respective class -> 60 frames of that respective video.

The data set is pretty small now; the total of all videos comes to 124. I will have to perform data augmentation to increase the number of videos. Say each class had 30-35 videos, then the total will be 330 – 385 videos… a bit decent, I guess. Which is why I am currently working on data augmentation.

As for the model implementation, I am thinking of 2 ways:

**Way 1:** Training a model with MediaPipe Keypoint data (This is numeric data with 3D coordinates of the keypoints).

I won’t be able to use a pretrained model for this but will have to train the model from scratch, for which I don’t have enough data. Let’s see about that though.

**Way 2:** Training a model with images.

I will be able to use a pretrained model for this approach. This method can further be split into 2 ways:

**Way A:** Training the model with MediaPipe Landmarks.

**Way B:** Training the model without MediaPipe Landmarks.

I need to try out both methods and not down the inferences. I don’t think we need to apply the face landmarks for this approach though.

To train the ResNet model, I need to resize the extracted frame to 224 x 224. To do that, I am cropping the center square from the frame and then resizing the images. This will make sure that the signer stays in the center of the resized frame and will retain the quality of the frames as the original aspect ratio. (Often when you resize the image to a smaller size, pixels are lost, and this affects the quality of the image… it makes it blurry).

So this is what will do, I will keep different folders for different tasks:

1. Contain original frames.
2. Contain total augmented frames.
3. Contain keypoints (NumPy arrays). [required for Way 1]
4. Contain images with keypoint landmarks.
5. Contain resized images with keypoints.
6. Contain resized images without keypoints.

Right now, I am searching for ways of augmenting video data.

**Production Log 2: (Sunday, 04/02/2024, 4:33pm)**

Okayy, so I’m writing this log from the computer lab of my university. Victor and Mujtaba are here too. This is what I have done so far:

* Data augmentation
* Resizing the frames to 224 x 224
* Applying media pipe to the augmented and resized images
* Labelling and creating a dataset from video frames to train the models.
* Written code for
  + Pretrained ResNet50 + BiLSTM
  + CNN + LSTM
  + Pretrained Xception + BiLSTM

I want to use the GPU here so I am downloading CUDA toolkit currently on the uni comp. Uni closes at 5 and its showing 3 hours of download time… well let’s see. I won’t log out of my account but I’ll lock the pc.

Back to the progress,

For Data augmentation, if have transformed the images in the following ways:

1. Turned them into black and white.
2. Increase the saturation and flipped the images.
3. Flipped the images.
4. I have another batch of rotated images (5 degrees) but I am not using them currently. I will use them later while training, I guess.

As for the resizing the images, I first cropped the center square (basically cropping out the extra background margins) and then resized them to 224 x 224 pixels so that they can be passed as an input to pretrained models such as Resnet 50, which were trained on the ImageNet dataset which contained images of the same size. I used the PIL library for image augmentation and resizing.

**Original Frame:**

A person in a blue dress

Description automatically generated

[1920 x 1080 pixel image]

**Resized Frame:** **MediaPipe Holistic applied frame:**

A person in a blue dress

Description automatically generated A person with lines drawn on her face

Description automatically generated

[224 x 224 pixel image] [Frame with pose and hand landmarks]

**Image augmentation:**

A person in a blue shirt

Description automatically generated A person in a dress

Description automatically generated

A person in a blue shirt

Description automatically generated A person in a blue shirt

Description automatically generated

[Clockwise direction: Saturation of 1.5, Contrast of 0, Left rotation of 5 degrees, Frame Flip]

After resizing the images, I applied MediaPipe holistic (pose and hand landmarks) to each of them and stored the keypoints in numpy format (which contains the extracted facial, pose and hand landmarks) in another folder. I used the MediaPipe library to apply MediaPipe Holistic.

As for creating the dataset out of the video frames, each image of a video of an action was read in numpy format using cv2, converted to RGB format (since cv2 reads the images in BGR format), and the pixel values were normalized. Each video is represented by 60 frames and the number of videos per action range from 36 to 40 videos. These pixel values of the images serve as the features of the dataset and the actions (classes) form the target column of the dataset.

The structure of the dataset is as follows:

**[[[[**R, G, B values of a frame**]**, [], [] … x 60 such frames**]**, [], [] … x 36 to 40 such videos of a class**]**,[], [] … videos of all classes**]**

So, each video (of 60 frames) constructs a row of the dataset.

There is an **alternate way** to create a dataset where each frame of a video serves as a row in the dataset. The dataset is then converted to and stored in csv format. I haven’t tried this method yet.

Then comes the stage of building the models. I have written code for 3 hybrid models so far by combining the codes found online. The models constructed so far are:

1. Pretrained ResNet50 + BiLSTM

Here ResNet 50 is a CNN model which is pretrained on the ImageNet dataset which contains 1000 classes.

1. CNN + LSTM
2. Pretrained Xception + BiLSTM

Here the Xception is a CNN model which is pretrained on the ImageNet dataset.

I have written code to generate a small section of the dataset to train the models on for initial testing purposes. The dataset contains 3 classes – police, follow and child, containing 40, 36 and 36 videos each. I have made use of the TensorFlow library and modules to develop hybrid models.

Currently I am in the process of setting up the environment to use GPU on the university computer to accelerate the training time of the models.