

**Predicting Road Crossing Behaviour using Pose Detection**

**(Under Guidance of Dr. Subhashish DasGupta)**

By:

Agniva Roy (C22001)

Ayan Mullick (C22006)

Ishaan Manchanda (C22011)

Navam Pradhan (C22016)

Preetam Saha (C22018)

Sk. Ahtesham Hussain (C22024)

# Abstract

Autonomous driving is one of the very complex technologies that is highly in demand. But safety will always be the top priority (both for the person inside the vehicle and people in the street) as accidents do not come along with a warning. Here, in this project our main objective is to predict the behaviour of a person on whether that person is going to cross the road or not. For the data collection, 62 videos were captured out of which 60 videos were for training and 2 videos were for testing. The videos were of two types, one being a person standing and not crossing the road and other one where the person is crossing the road. Based on these data, two sequential models (1. LSTM, 2.GRU) have been used to predict the crossing behaviour of test samples (one being crossing and the other being not crossing).

The training dataset was generated from the training videos using OpenCV (developed by Intel) and MediaPipe library (developed by Google). Each video is composed of multiple frames where each frame generated 33 pairs of pose points by MediaPipe. The frames were labelled manually as 0 (when the person is not crossing) and 1 (when the person is crossing). For training the model, a sequence length of 15 was taken and the dataset was modified by shifting frames one by one in a sliding window fashion.

The trained model was then used to predict whether a person is about to cross the road or not.

Then we used this dataset for LSTM and GRU models to predict whether a person is going to cross or not based on probability values.

# Acknowledgement

We would like to thank Dr. Subhashish Dasgupta who had continuously been touch with us during the entire project, for the support and guidance in completing the project.

Also, we would like to express our gratitude to all of our group members. The project would not have been successful without their cooperation and inputs.

Additionally, we would like to thank our classmates of DS-Jan’22 batch for volunteering to give their videos for training our model.

Table of Contents

[Abstract 2](#_Toc117190257)

[Acknowledgement 4](#_Toc117190258)

[Introduction 6](#_Toc117190259)

[MediaPipe 7](#_Toc117190260)

[LSTM 8](#_Toc117190261)

[Gated Recurring Unit (GRU) 9](#_Toc117190262)

[Literature Review 10](#_Toc117190263)

[Methodology 12](#_Toc117190264)

[Data Collection 12](#_Toc117190265)

[Labelling 14](#_Toc117190266)

[Training the Model 15](#_Toc117190267)

[Design 18](#_Toc117190268)

[Analysis 18](#_Toc117190269)

[Findings 20](#_Toc117190270)

[Conclusions 24](#_Toc117190271)

[Recommendations and Future Scopes: 25](#_Toc117190272)

[References 26](#_Toc117190273)

# Introduction

Autonomous driving is one of highly researched topics in the present time. There are already self-driving cars in road, but they still require a human presence to function properly. There are many developments still undergoing in this field. But the part where the autonomous driving is still lacking is predicting the behaviour of an object beforehand in order to prevent a disaster.

The objective of this study is to address human pose recognition from video sequences by formulating it as classification problem. The goal of the work is to develop an end-to-end sequential model that tries to solve the problem of road crossing behaviour of individuals.

The sequential model trained can be used to detect the behavioural pose of the individuals and to take decisions on the fly by predicting whether a person is going to cross the road or not.

Human pose estimation aims at predicting the poses of human body parts and joints in images or videos. Since pose motions are often driven by some specific human actions, knowing the body pose of a human is critical for action recognition. Pose estimation operates by finding key points of a person or object. Taking a person, for example, the key points would be joints like the elbow, knees, wrists etc.

Single pose estimation is used to estimate the poses of a single object

Human pose estimation is a challenging task as the body’s appearance changes dynamically due to diverse forms of clothes, arbitrary occlusion, occlusions due to the viewing angle, and background contexts.

## MediaPipe

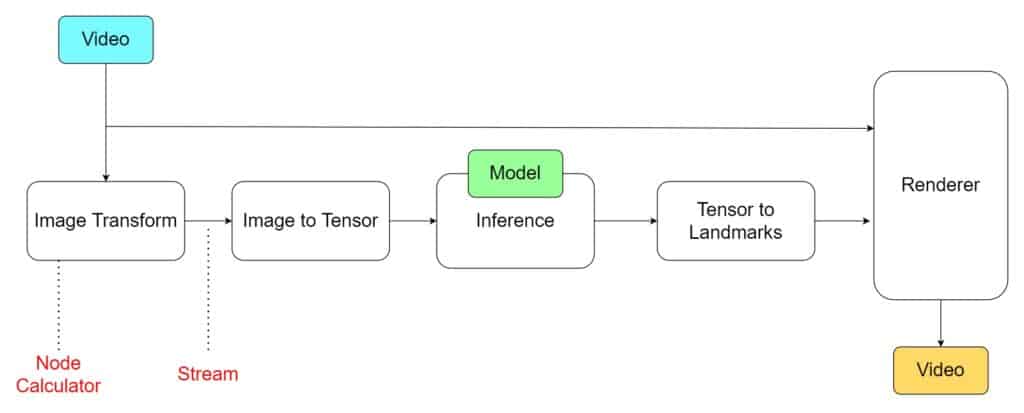
MediaPipe offers cross-platform, customizable ML solutions for live and streaming media.

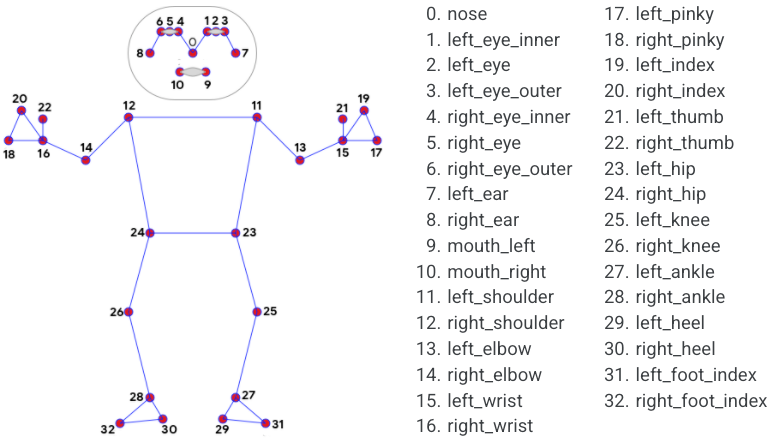
End-to-End acceleration: Built-in fast ML inference and processing accelerated even on common hardware.

Build once, deploy anywhere: Unified solution works across Android, iOS, desktop/cloud, web and IoT.

Ready-to-use solutions: Cutting-edge ML solutions demonstrating full power of the framework.

Free and open source: Framework and solutions both under Apache 2.0, fully extensible and customizable.





Source: <https://google.github.io/mediapipe/solutions/pose.html>

## LSTM

LSTM networks are capable of learning-order dependence in sequence-prediction problems. LSTM has a compelling feature of not getting stuck by vanishing gradient problem. The additional layers have both sigmoid and tanh activation functions to control the inputs.

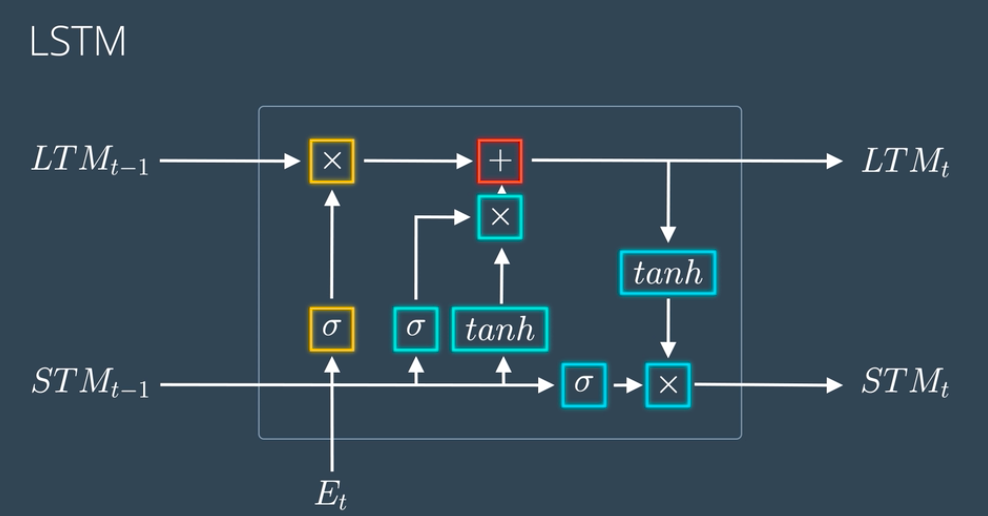


Figure: Schematic Diagram of LSTM Architecture

Source: <https://www.analyticsvidhya.com/blog/2021/01/understanding-architecture-of-lstm/>

## Gated Recurring Unit (GRU)

GRUs are very similar to Long Short-Term Memory (LSTM). Just like LSTM, GRU uses gates to control the flow of information. They are relatively new as compared to LSTM. This is the reason they offer some improvement over LSTM and have simpler architecture.

Another Interesting thing about GRU is that, unlike LSTM, it does not have a separate cell state (Ct). It only has a hidden state (Ht). Due to the simpler architecture, GRUs are faster to train.

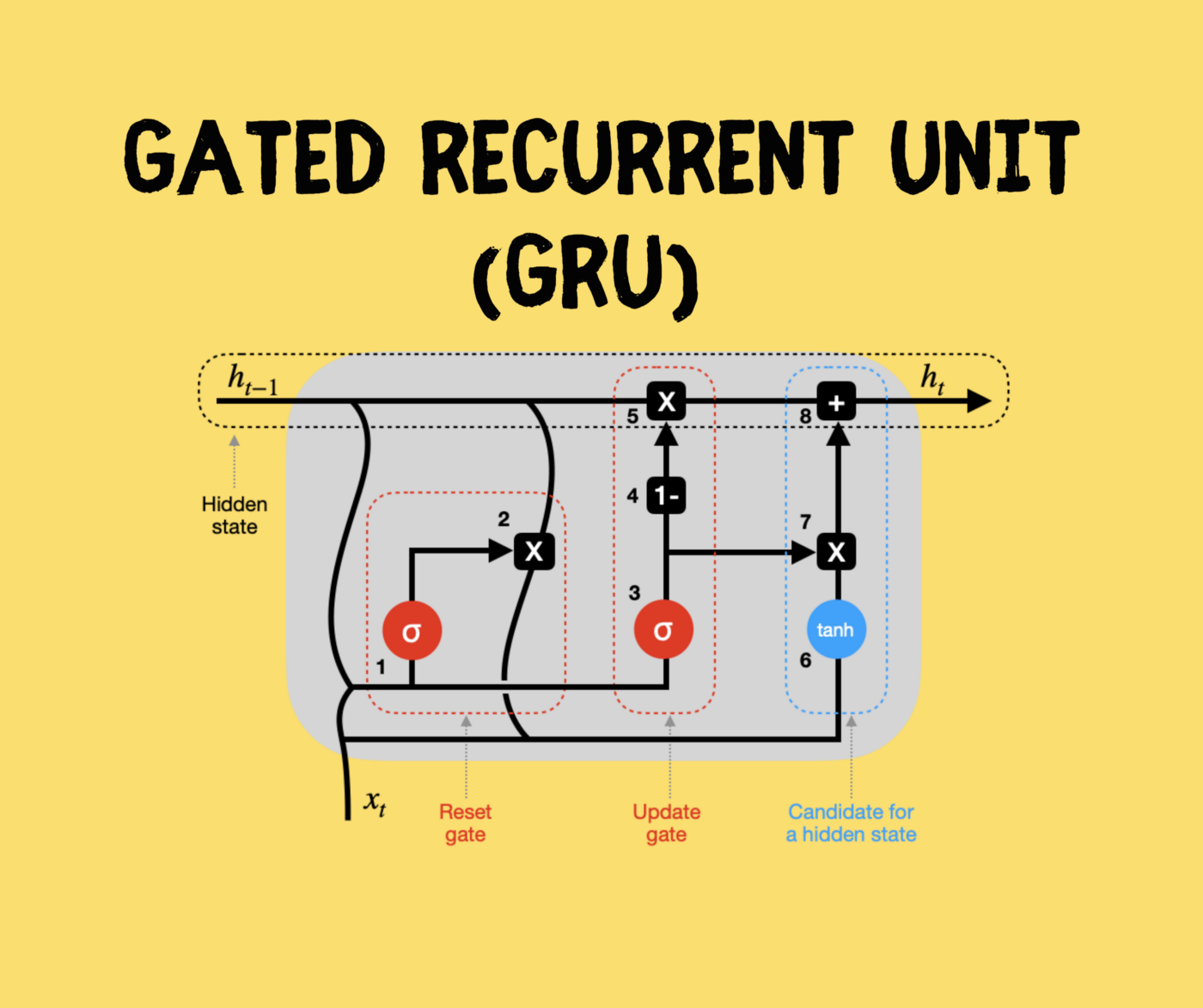


Figure: Schematic Diagram of GRU Architecture

Source: <https://towardsdatascience.com/gru-recurrent-neural-networks-a-smart-way-to-predict-sequences-in-python-80864e4fe9f6>

# Literature Review

Human pose estimation has been extensively studied in computer vision research area. There is extensive literature on pose estimation (Andriluka et al., 2009) and object recognition. Early works on single-image pose estimation started from building graphical structures to model the relations between joints (Felzenszwalb & Huttenlocher, 2005). The performance of these methods has recently been surpassed by CNN based methods (Wei et al., 2016). Those deep models had the capacity to generalize from unseen scenes by learning various spatial relations from data. Previous work has been done using MediaPipe to understand the relationship between the class and coordinates to classify and detect custom body languages (Singh et al., 2022). Previous work on pose detection has been done using Random Forest (Rogez et al., 2008). Our contributions are as follows (1) using MediaPipe to determine key points (2) manually label the frames from the movement of the subject (3) train LSTM or GRU model to learn human body language for pose estimation.

LSTM have been widely used in pose estimation tasks such as motion tracking, action recognition and Multi-Person pose estimation (Li et al., 2019). There has also been use of p-LSTMs (K. Lee et al., 2018), their findings saw an improvement in accuracy of 11.2%. GRU has also seen an increase in adoption, since it is a simplified version of LSTM, for example, 3D Hand Pose estimation (Guo et al., 2020) and Fall Detection based on pose estimation (Kang et al., 2021). There have been several studies carried out to understand the pedestrian behaviour, which are influenced by different factors such as pedestrian perception, roadway, and environmental characteristics. There has been work done regarding traffic flow and road-crossing behaviour of pedestrians (Jin et al., 2013) and illegal road crossing behaviour of pedestrians (Demiroz et al., 2015). Our work involves predicting the road crossing behaviour of pedestrians, which could then be used as an accident preventing software in Self-driving cars.

# Methodology

For our project there were no source of any secondary data. So, the data had to be primary data which were collected in the form of videos. The video length was of approx. 8-12 secs.

## Data Collection

The videos are of two categories – a person crossing the road and a person not crossing the road. Care was taken to see that there were approximately the same number of videos of a person crossing from left to right as well as from right to left.

Also, the balance was maintained between the videos of person not crossing and person crossing the road.

After this we labelled the frames.

Firstly, we tried to extract key points using MMCV but the key points we got from it was only 17 based on Coco topology. Also, this pre-trained was very difficult to interpret and not that flexible to change any codes.



MMCV pose detection output

Whereas, MediaPipe gives 33 key points based on Blazepose topology. Furthermore, MediaPipe is easy to interpret and flexible to change the codes. Therefore, we used MediaPipe.



MediaPipe pose detection output

## Labelling

**For labelling the frames of the videos for the person crossing the road:**

The video is played and the approximate time at which the person has started moving was noted. The videos collected were of 30 frames per second.

A labeller function “*labeller*” was created in which the video path and frame number was passed as input.

To read the video, OpenCV was used where each video is a sequential combination of frames. Each frame is an image. The image is read in BGR format and then it is converted to RGB format for MediaPipe input. The MediaPipe library is used to extract the 33 key pair points (x, y, z, score) corresponding to the 33 specific locations in a human body. For the analysis, only the x, y co-ordinates were required.

For labelling, if the frame number of the video is less than the frame number that was passed to the function, then the corresponding label is zero, else the label is one.

After all the frames have been captured, the corresponding points were returned along with labels in a list.

**For labelling the frames of the videos for the person not crossing the road:**

For the videos in which the person is not crossing the road, a function “*labellernotmoving*” was created. The working is almost the same as the “*labeller*” function. But in this case, all the frames were labelled as zero.

The procedure is repeated for all the 60 videos.

A list was obtained from either of the above-mentioned functions. Then this list is passed to a function named “*setdata*” to convert this list into a Pandas data frame which has 67 columns. The number of rows is equal to the number of frames. The same process is repeated for all the 60 videos.

This data frame is then saved to a file in csv format which will then be used for training the model.

## Training the Model

The csv files were read and saved in a list.

Sequential models require input to be in a specific format. The data must be in the form of rows, where each row contains a sequence of specific length, 15 in this case.

The data was transformed using “*generate\_data*” function from the original data.

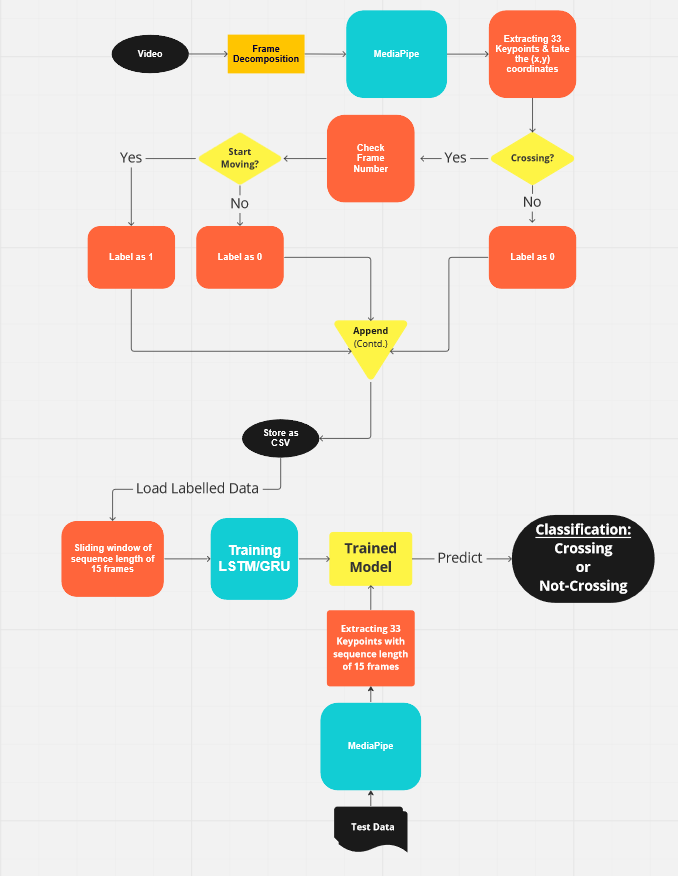
In a list, the sequence of frames and the corresponding outputs were stored, and that list is supplied to the LSTM model for training.

One layer of LSTM is used which has 50 hidden nodes and a dropout layer with probability 0.5, and one dense layer with activation function sigmoid to get the output between 0 and 1.

The model is compiled using Adam as optimizer, binary cross entropy as loss function, and accuracy as metric.

For training a batch size of 32 is taken and epoch of 10 for each video with a loop running 10 times with different permutations of different video sequences.

The same steps were followed for the GRU model as well.



Flow Chart

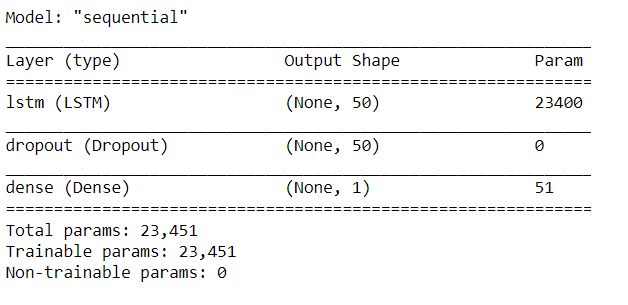
# Design

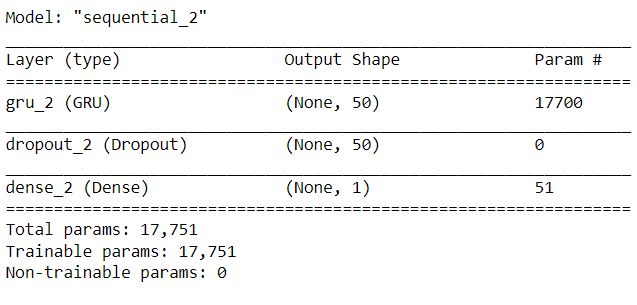
## Analysis

The LSTM model is trained, and the trained model is saved in a Saved Model format which creates a directory containing a protobuf binary and a TensorFlow checkpoint. A function “*test*” is created for testing the model with test video. In this function the video path of the test video is passed as input and x and y coordinates are extracted in the same manner using MediaPipe like it was used when labelling the training data. MediaPipe kept on collecting the key points till it arrived at 15th frame. If the sequence length is 15, it starts giving output as a probability score corresponding to the last frame of the sequence.

Thus, the LSTM model is tested for two new videos and the corresponding output is obtained on whether the person had a crossing behaviour or not.

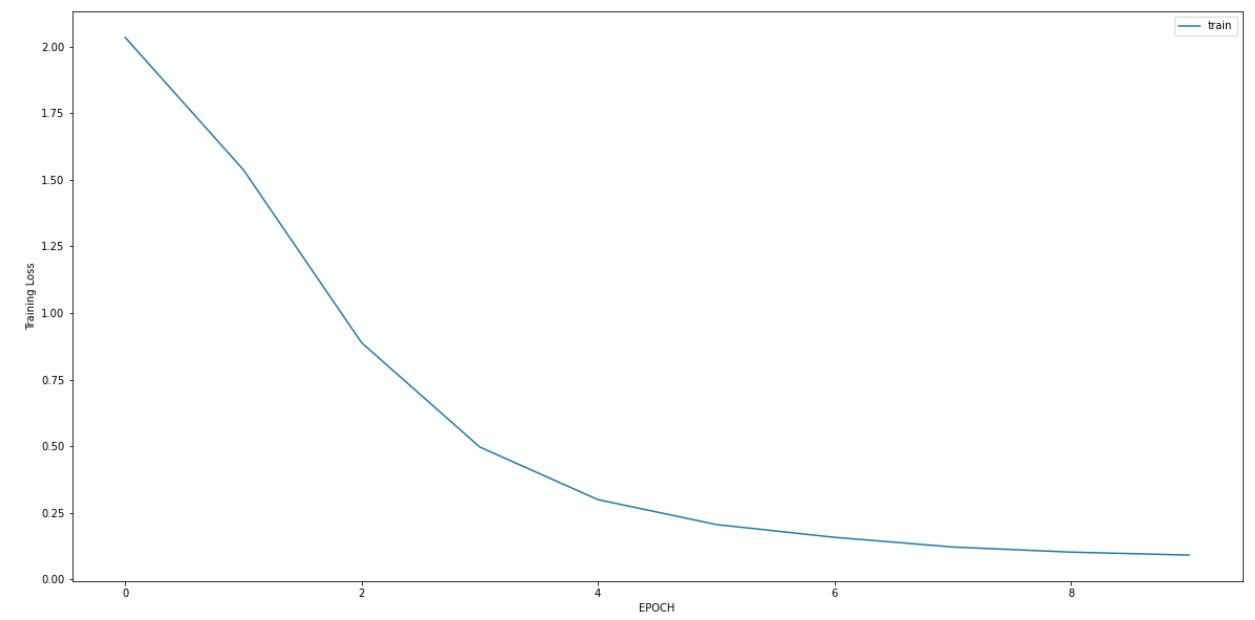
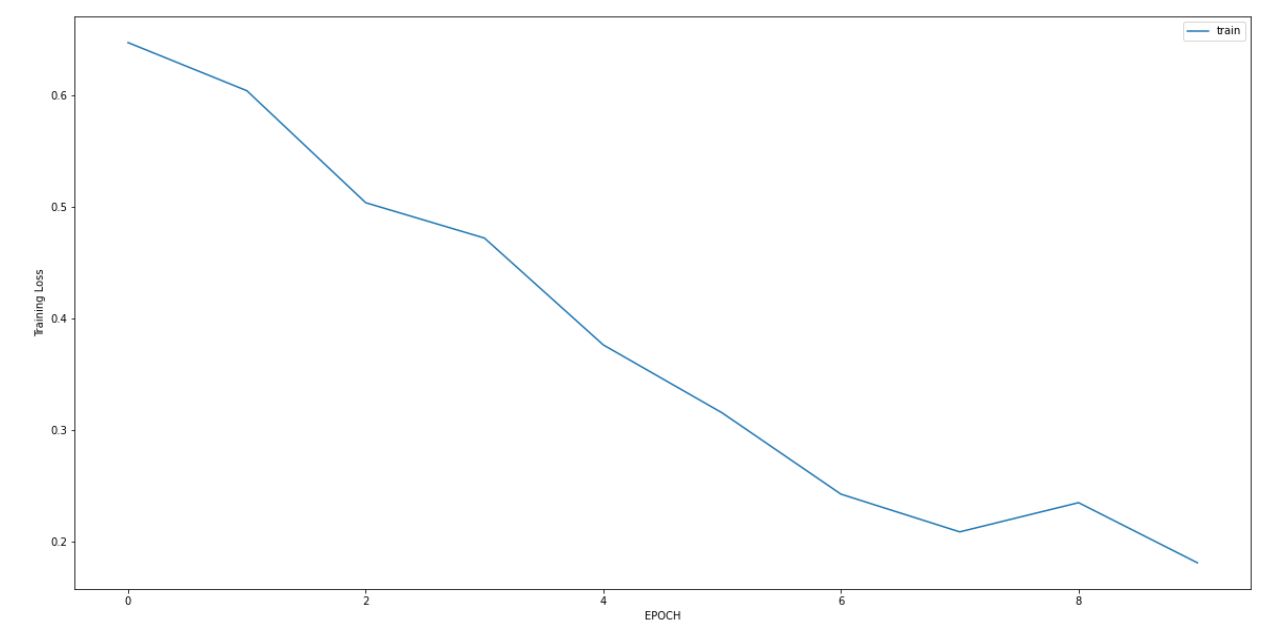
The same process is followed for GRU model as well and got the corresponding outputs.



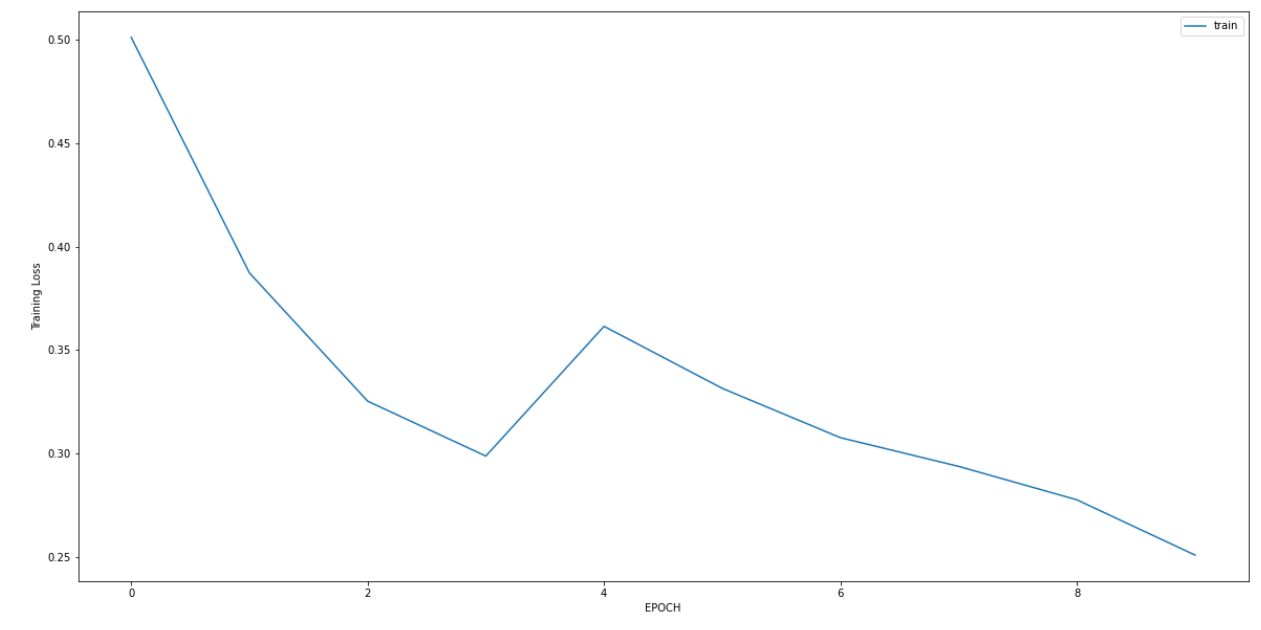
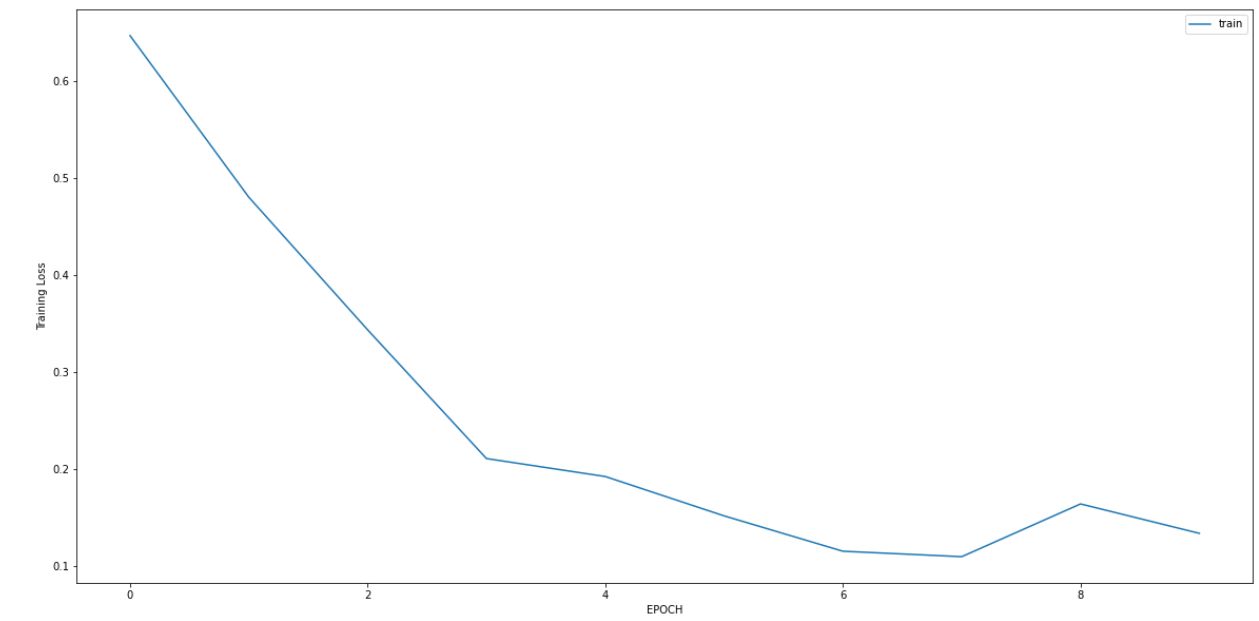


## Findings

The training losses is observed with increasing epochs for the corresponding videos. Out of all videos, some of them have been shown below:



LSTM – Training Loss over 10 Epoch



GRU Training Loss over 10 Epoch

It can be observed that overall training loss is decreasing as the number of epoch increases. However, in maximum number of videos, the decrease is very smooth while in some it is quite unstable with slight increase in training loss in some of the epochs.

In case of GRU, there is overall decrease of training loss with increase in the number of epochs but the decrease of loss is noticeably unstable compared to LSTM in comparatively a greater number of videos.

LSTM Output





GRU Output





# Conclusions

As we are moving towards automated technology era, self-driving cars are going to become more and more prevalent now-a-days. Keeping these things in mind, our project which has the objective to predict the road crossing behaviour of the person, will help to take decision in real time whether a person will cross the road or not which will eventually help in preventing many accidents. Here, we have used only 60 videos due to the shortage of time of the project, but to get more robust model we need a greater number of data points. Although GRU is faster than LSTM due to having lesser number of parameters, accuracy is better for LSTM than GRU. Also, the training loss which decreases with the increase of number of epochs is not that much stable for GRU vis-a-viz LSTM. This model can further be improved upon and used for Multi-Pose detection as well which is more realistic in real life. For more number of videos and more complex data, we can use stacked LSTM although we have tried it but it was giving an overfitted model. This model can be used as an input for reinforcement learning as well.

# Recommendations and Future Scopes:

* We can get more accurate results if we can provide more training dataset, have more computational power.
* We can extend this to multi-pose detection as well.
* This can be used not only in self-driving cars, also in normal cars to give additional warnings to prevent accidents due to carelessness.

# References

Andriluka, M., Roth, S., & Schiele, B. (2009). Pictorial structures revisited: People detection and articulated pose estimation. *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 1014–1021. https://doi.org/10.1109/CVPR.2009.5206754

Demiroz, Y. I., Onelcin, P., & Alver, Y. (2015). Illegal road crossing behavior of pedestrians at overpass locations: Factors affecting gap acceptance, crossing times and overpass use. *Accident Analysis & Prevention*, *80*, 220–228.

Felzenszwalb, P. F., & Huttenlocher, D. P. (2005). Pictorial Structures for Object Recognition. *International Journal of Computer Vision*, *61*(1), 55–79. https://doi.org/10.1023/B:VISI.0000042934.15159.49

Guo, F., He, Z., Zhang, S., Zhao, X., & Tan, J. (2020). Attention-Based Pose Sequence Machine for 3D Hand Pose Estimation. *IEEE Access*, *8*, 18258–18269. https://doi.org/10.1109/ACCESS.2020.2968361

Jin, S., Qu, X., Xu, C., & Wang, D.-H. (2013). Dynamic characteristics of traffic flow with consideration of pedestrians’ road-crossing behavior. *Physica A: Statistical Mechanics and Its Applications*, *392*(18), 3881–3890.

Kang, Y., Kang, H., & Kim, J. (2021). Fall Detection Method Based on Pose Estimation Using GRU. In R. Lee & J. B. Kim (Eds.), *Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing* (pp. 169–179). Springer International Publishing. https://doi.org/10.1007/978-3-030-67008-5\_14

Lee, K., Lee, I., & Lee, S. (2018). Propagating LSTM: 3D Pose Estimation Based on Joint Interdependency. In V. Ferrari, M. Hebert, C. Sminchisescu, & Y. Weiss (Eds.), *Computer Vision – ECCV 2018* (pp. 123–141). Springer International Publishing.

Li, M., Zhou, Z., & Liu, X. (2019). Multi-Person Pose Estimation Using Bounding Box Constraint and LSTM. *IEEE Transactions on Multimedia*, *21*(10), 2653–2663. https://doi.org/10.1109/TMM.2019.2903455

Rogez, G., Rihan, J., Ramalingam, S., Orrite, C., & Torr, P. H. S. (2008). Randomized trees for human pose detection. *2008 IEEE Conference on Computer Vision and Pattern Recognition*, 1–8.

Singh, A. K., Kumbhare, V. A., & Arthi, K. (2022). Real-Time Human Pose Detection and Recognition Using MediaPipe. In V. S. Reddy, V. K. Prasad, J. Wang, & K. T. v Reddy (Eds.), *Soft Computing and Signal Processing* (pp. 145–154). Springer Nature Singapore.

Wei, S.-E., Ramakrishna, V., Kanade, T., & Sheikh, Y. (2016). Convolutional pose machines. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 4724–4732.