

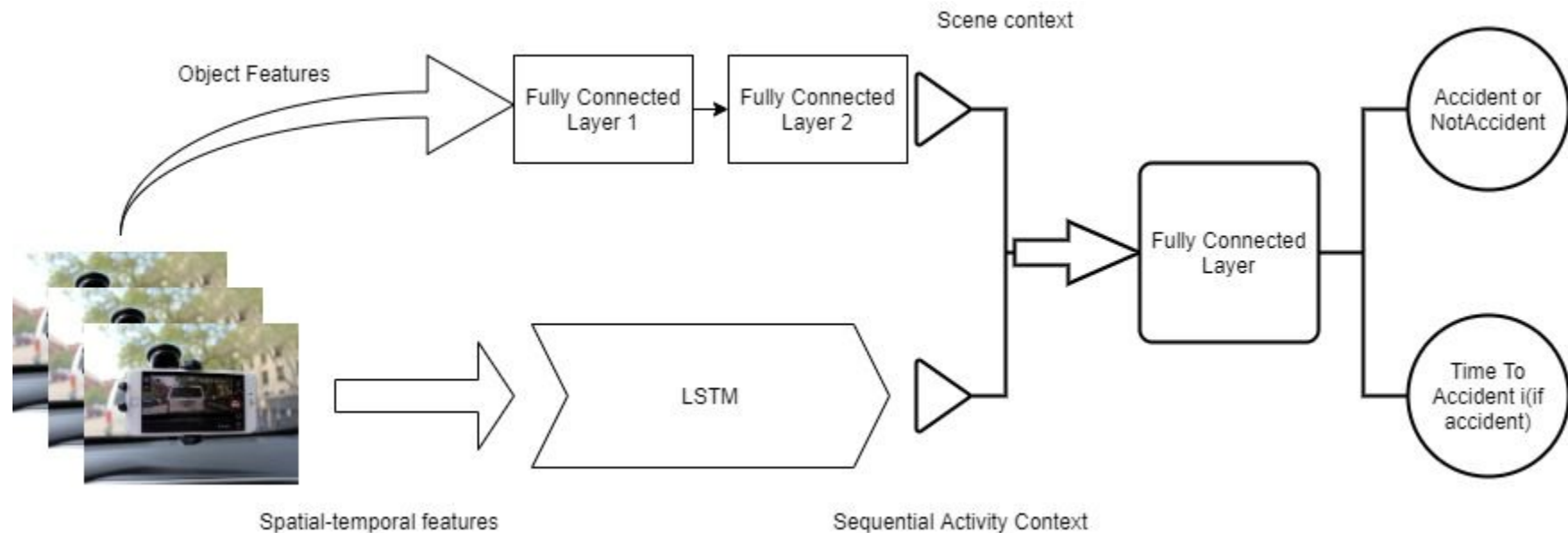



Final project Ideas CSCI-631

Team 9

- Tejas Arya (ta2763)
- Amritha Venkataramana (axv3602)

Model Architecture





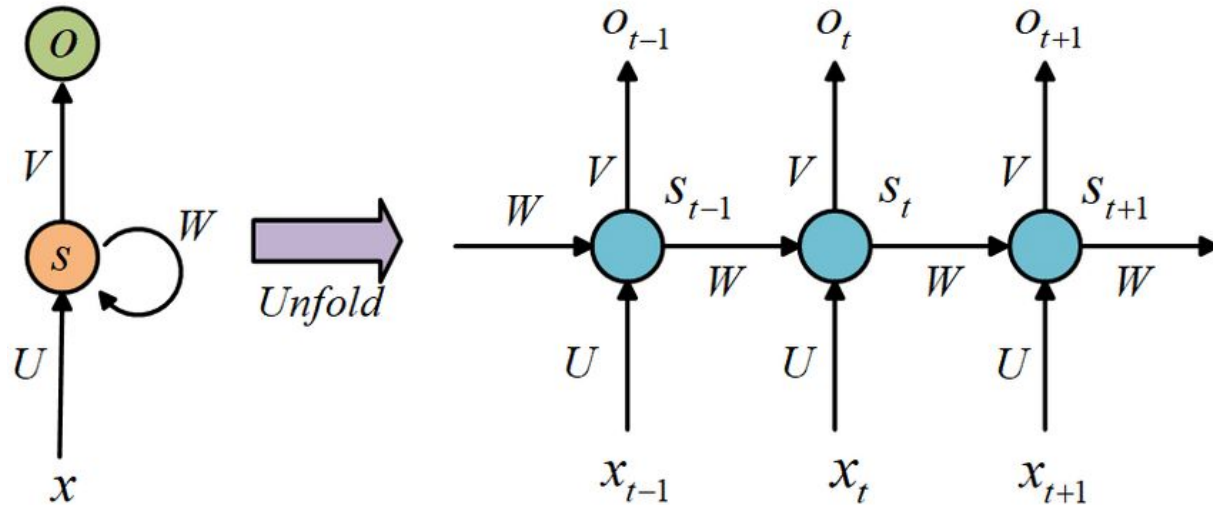
LOSS FUNCTION: A loss function is used in machine learning models in order to understand and evaluate the performance level of the algorithms used in modeling the dataset. In our situation, lower value of a loss function implies good results while a larger value indicates that there are discrepancies in the predictions. For classification tasks, one of the most commonly used loss functions for classification algorithms is the Cross Entropy function.


1. Cross Entropy (For label prediction): This function calculates difference between the probability distributions. It ensures that there is minimum difference between these two values. For best prediction, cross entropy holds a value of 0.
2. Root Mean Squared Error (For Time to Accident (if LABEL == ACCIDENT)): RMSE penalizes errors heavily by squaring the errors.

OPTIMIZING THE NETWORK: After studying various research works done in the same area that requires predicting appropriate frames, we decided to go with optimizing the network with ADAM optimizer.

ADAM optimizer is a state of the art, adaptive learning rate method which can be seen as combining the best properties of RMSProp and Stochastic Gradient Descent.

Hierarchical Recurrent Neural Network



- 
- 1) HRNN is a collection of multiple recurrent neural networks(RNNs) stacked one over the other in order to model hierarchical structures in sequential data like images, video frames etc.
 - 2) The key idea behind this lies in updating the weights of neurons belonging to different layers of stack corresponding to the layers of hierarchies in the data.
 - 3) Stacked RNNs construct hidden states from two different states, one being the previous state and the other being the current state.
 - 4) Stacking RNNs allows storing more information in the hidden layers and this ensures more information in the layers close to the output layers.
 - 5) For classification purposes especially involving binary classification, HRNN can be most effective.



Machines used:

1. Dell G7; 16GB RAM, 6GB NVIDIA GeForce 1060 GPU
2. Acer Predator Helios; 16GB RAM, 8243MB Intel(R) UHD Graphics 360
3. Google Colab GPU

Challenges faced



- 1) While understanding the working of the project code, lots of functions and libraries used are in the TensorFlow 1 version. However, TensorFlow1 was deprecated and making that migration to Version 2 to tweak the libraries into working a certain way as required for the project has been a bit challenging. However, we are progressing on understanding what best suits the model and its prediction to incorporate the necessary changes.
- 2) Obtaining the dataset for the paper “Anticipating Accidents in Dashcam Videos”, on which our project was originally based, was quite difficult. We reached out to the authors and could obtain the dataset after quite a while.
- 3) TensorFlow 2 is relatively new, and new functionalities are being added everyday. Moreover, TensorFlow contains vast amount of methods and pipelines. It took us quite a while to get familiar with its methods. We currently, are in the middle of the implementation phase and trying to work out the errors. Several of them included “Deprecation Warnings” which is currently majorly handled.
- 4) Coordinating the coding and research while maintaining the social distance was a challenge. Usually, it helps while coding, when you can point out each others errors which can be blindsided by one person. However, we ensured we were following a strict timeline to work on the project and pointing out each others’ errors and guiding each other through Zoom screen sharing. Nonetheless, all the errors have not been resolved and all the functionality has not been completed and it is still a work in progress.



Future Work

After the completion of this project, we would like to continue the research in the direction of integrating the trained model with the hardware using IoT devices with live footage for testing instead of stored clips.

Moreover, what can be done is to actually let the machine which stores our model continuously learn from the live footage and predict when to stop in case an accident is going to occur simultaneously.

However, this needs to be done in a controlled environment for us to make sure there's no unethical testing and danger to any individual, animals or property.

Test case could be throwing random obstacles in way of a running car and simulating actual miniature road conditions on a toy car fitted with our model.



References

1. Fu-Hsiang Chan, Yu-Ting Chen, Yu Xiang, Min Sun: Anticipating Accidents in Dashcam Videos
https://yuxng.github.io/chan_accv16.pdf
2. Tahmida Mahmud , Mahmudul Hasan , Amit K. Roy-Chowdhury: Joint Prediction of Activity Labels and Starting Times in Untrimmed Videos
http://openaccess.thecvf.com/content_ICCV_2017/papers/Mahmud_Joint_Prediction_of_ICCV_2017_paper.pdf
3. HRNN model reference https://github.com/rwk506/CrashCatcher/blob/master/HRNN_training.md
<https://github.com/ankush17100/Accident-Detection-Using-Deep-Learning/tree/master/Accident%20Detection>