Report on Real-time Intent Prediction of Pedestrians for Autonomous Ground Vehicles via Spatio-Temporal DenseNet

Rahavee Prabakaran

rprabak1@asu.edu[1]

I. Introduction

The problem addressed in the chosen reference [1] is pedestrian intention prediction in a variety of urban environments. The problem is simple for a person to interpret and infer whether or not the pedestrian is intending to cross, but it is not so simple for Automatic Ground Vehicles (AGV) to forecast the pedestrian's intention [2]. For instance, in the scenario depicted in Figure 1, any human motorist would assume that the pedestrian in the image is attempting to cross. On the other hand, an autonomous vehicle (AV) could have a hard time anticipating this behavior [3][4]. As a result, one of the most important qualities for AGVs to gain is the ability to understand human actions and intents. The difficulty in tackling the challenge stems from the fact that models must continually watch pedestrians and forecast their intentions. The importance of resolving the issue rests in the total autonomy of AGVs as well as the safety of other road users such as pedestrians and passengers within AGVs. The picture sequences from a monocular RGB camera are used to solve the problem of pedestrian intent action prediction in urban traffic scenarios. Based on a tracking-by-detection approach and a unique Spatio-Temporal DenseNet (ST-DenseNet) model, the real-time framework can effectively identify, track, and anticipate the intended activities of pedestrians in the field of computer vision, this area of study is still quite active. Although the perception job has received a lot of attention, it still offers a lot of room for excellent study. Complete autonomy of AGVs is achieved by combining several tasks such as best directions to the target, environmental localization, speed control, lane change decision, and object and semantic detection. The probable trajectory of pedestrians is likewise seen as a significant issue that is currently being investigated. This reference makes an attempt to resolve the issue.

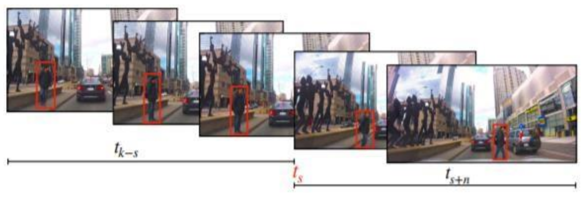


Fig 1. Crossing scenario from the newly released JAAD dataset.

II. Data Preprocessing

A. Approach to the solution

An approach to the problem statement as described in the paper is building a novel real-time unified and flexible fframework for prediction and localization of a pedestrian's intention from video sequences. Here the concept of Densely convolutional networks is used to model the Spatio-Temporal information in the video sequences. In recent times Spatio-Temporal ConvNet have shown striking results and reliable temporal modelling especially where image sequences from an RGB camera come into play and in fact the results are even better than that of RNNs. As far as DenseNets are concerned, they have shown remarkable results while dealing with modeling spatial information in deep 2D Convents when a smaller number of parameters are to be trained.

To give a better understanding about the proposed real-time unified framework, there is a sub-module integrated for real-time tracking-by-detection of pedestrians from video sequences based on YOLOv3 and Simple Online and Realtime Tracking (SORT) algorithm using Unscented Kalman Filter (UKF) as shown in Fig 2. This integrated sub-module takes into account the uncertainty that exists in noisy observations and detections from an observer who is moving.

B. JAAD Dataset

The dataset used to train the models is the Joint Attention for Autonomous Driving (JAAD) dataset [5]. It is a vehicle-based dataset captured using a dash camera (at 30 FPS with 1080H x 1920W). The dataset is focused on the behaviors of drivers and pedestrians while crossing and the factors that influence them. It has a collection of 346 video clips (5-10 seconds long) taken from 240 hours of driving footage. The dataset has proven to be suitable for pedestrian detection because of the bounding boxes with occlusion tags are provided for every pedestrian. Each short video clip has associated tags, timestamped behavior labels, demographic attributes and visible traffic scene elements.

C. Preprocessing for ST-DenseNet

The first phase of training our proposed framework for the pedestrian intent prediction task is training ST-DenseNet. The labels such as Crossing or not, which describes are the pedestrian's intent action is done by taking all the k frames before start of the activity as the input image sequence. For labels which describe the pedestrian's intention to cross, since all the Bounding Boxes (BBoxes) sequences from the JAAD Dataset comes with size 1080H×1920W, it is important all the BBoxes sequences are cropped and resized to 100H×100W.

This whole sequence of frames is labeled as intended crossing action. For labels which describe the pedestrian's intention to non- crossing, the instances of pedestrians standing or walking beside curb are chose and the corresponding BBoxes are cropped and resized to 100H×100W. Further pre-processing is done so that the prediction of the pedestrian's intended action is achieved at least half a second before the action is completed.

Note: All the references are hyperlinked directly while used in the literature.

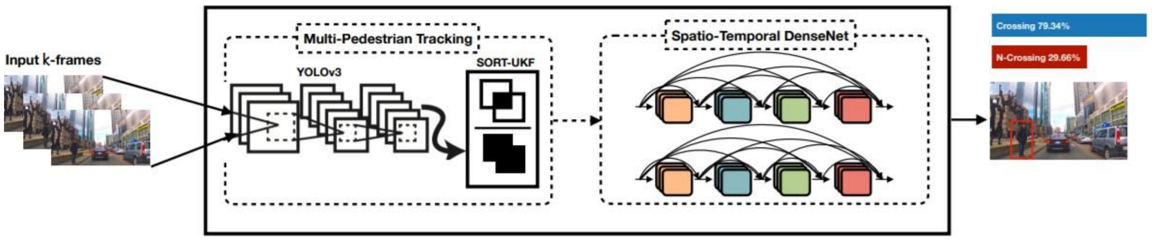


Fig 2. Unified Framework for Intent Action Prediction of Pedestrians. The input is the sequence of k - consecutive images and output is probability of crossing or not.

The dataset is split into 70% training, 15% for validation and 15% for testing. As an output of the preprocessing, 3602 sequence samples (16 BBoxes each) with 3061 for training and validation; 541 for testing are obtained.

D. *Data Processing for Multiple Object Tracking Stage*

For training the YOLOv3, transfer learning feature of Deep ConvNet is used to enhance model capabilities. The YOLOv3 model initially trained on MS COCO dataset is fine-tuned using the training and validation split of the JAAD dataset. As part of preprocessing, raw images of training (70% of the dataset) are obtained and resized them to (416H × 416W) to comply with the original YOLOv3 model. The output thus obtained is 10K images annotated with at least one pedestrian BBox for each image. The output bounding boxes (BBoxes) from ConvNet-based object detection model (YOLOv3 model) are given to Simple Online and Realtime Tracking (SORT). SORT has achieved state-of-the-art results in Multiple Object Tracking (MOT) benchmarks.

One of the main advantages of the SORT is its super real-time performance, where it was benchmarked as 20x faster than the other state-of-the-art trackers. It provides frame to frame associations which makes it more efficient. From the output of the YOLOv3 model, SORT predicts the next time step motion of the BBoxes using linear Kalman Filter (KF). In this real-time framework, SORT-UKF is used, which is an advancement while dealing with traffic environments where pedestrians move in nonlinear motion as shown below. UKF is Unscented Kalman Filter (UKF) which deals will non-linear motion estimation model for the multi-pedestrian tracking stage. It assumes constant velocity models for the pedestrians and each detected BBox of pedestrian in the scene is associated based on IOU with the predicted BBoxes from the UKF. Then, the detections are used for updating the UKF.

III. DESCRIPTION OF NETWORK ARCHITECTURE

To solve the problem of pedestrian crossing, two networks are used in cascaded nature. The first network is the well-known YOLOv3 and the second is the Spatio-Temporal DenseNet (ST-DenseNet).

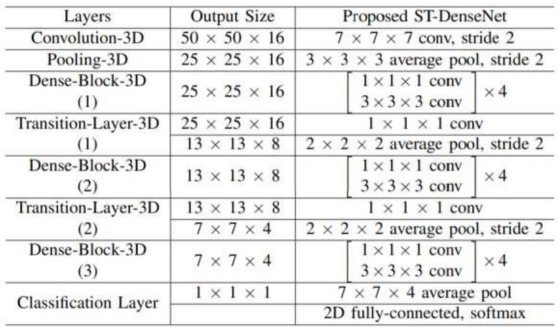
A. *Multi-Pedestrian Tracking*

The first stage is called Multi-Pedestrian Tracking. YOLOv3 (You Only Look Once version 3) is one of the real time object detection models that detects different objects from live feed, images or video files. YOLO is developed by Joseph Redmon and Ali Farhadi[6]. Of all the 80 classes it can detect, we are only interested in pedestrians (humans). YOLO predicts BBoxes i.e., 4 values namely, ***bx,by,bw,bh***. During training, YOLOv3 uses sum of squared losses. It uses Logistic Regression to detect the objectness of a detection. For Object classification, independent logistic classifiers and binary cross-entropy are used to train the model. This allows the model to detect overlapping labels like "man" and "person" in Open Image Dataset (OID)[7].

A new network is introduced as feature extractor for the YOLOv3. The feature extractor used has 3 x 3 and 1 x 1 convolution layers in successive format. It has total of 53 layers, so it is called Darknet-53. YOLOv3 detects the pedestrian and the bounding boxes are given to SORT UKF (Simple Online Realtime Tracking Unscented Kalman Filter) to perform multi object tracking simultaneously [8]. SORT UKF gives the output based on the bounding box co-coordinates provided by YOLOv3. SORT UKF gives an estimated state of each target by

$$x = [u,v,s,r,\dot{u},\dot{v},\dot{s}]^T,$$

where *u* and *v* represent the horizontal and vertical pixel location of the center of the target, while the scale *s* and *r* represent the scale (area) and the aspect ratio of the target's bounding box respectively. The dot above variables shows the velocity state estimation by SORT-UKF.



Layers	Output Size	Proposed ST-DenseNet
Convolution-3D	$50 \times 50 \times 16$	$7 \times 7 \times 7$ conv, stride 2
Pooling-3D	$25 \times 25 \times 16$	$3 \times 3 \times 3$ average pool, stride 2
Dense-Block-3D (1)	$25 \times 25 \times 16$	$\begin{bmatrix} 1\times1\times1 \text{ conv} \\ 3\times3\times3 \text{ conv} \end{bmatrix} \times 4$
Transition-Layer-3D (1)	$25 \times 25 \times 16$	$1 \times 1 \times 1$ conv
	$13 \times 13 \times 8$	$2 \times 2 \times 2$ average pool, stride 2
Dense-Block-3D (2)	$13 \times 13 \times 8$	$\begin{bmatrix} 1\times1\times1 \text{ conv} \\ 3\times3\times3 \text{ conv} \end{bmatrix} \times 4$
Transition-Layer-3D (2)	$13 \times 13 \times 8$	$1 \times 1 \times 1$ conv
	$7 \times 7 \times 4$	$2 \times 2 \times 2$ average pool, stride 2
Dense-Block-3D (3)	$7 \times 7 \times 4$	$\begin{bmatrix} 1\times1\times1 \text{ conv} \\ 3\times3\times3 \text{ conv} \end{bmatrix} \times 4$
Classification Layer	$1 \times 1 \times 1$	$7 \times 7 \times 4$ average pool
		2D fully-connected, softmax

Fig 3. ST-DenseNet Architecture with *k=24* as growth parameter.

B. *ST-DenseNet*

The next network in the architecture used is ST-DenseNet which is as shown in Fig 3. It is developed by introducing 3D kernels for convolutions and pooling layers in contrast to 2D kernels used in original DenseNet[9]. 3D convolution layers advocate the convolution of its feature maps spatially (as in the case of 2D kernels) and not leaving out the temporal dependency between the consecutive frames. Analogously, the 3D pooling layers reduce the size of feature maps both spatially and temporally. The kernel of both 3D convolution layers and 3D pooling layers is of size (S × S × D), where S is the spatial size and D is the input video frames depth/length. The main feature of DenseNet architecture lies in dense connections between each layer and its preceding layers. This helps to improve the information flow and gradients which helps to accelerate the training phase significantly. It also reduces the number of parameters because the networks learn to preserve information and eliminate the redundancy in re-learning of the same weights repeatedly.

Each Conv net in ST-DenseNet as shown in architecture is composed of three consecutive operations namely, Batch Normalization (BN)[10], ReLU, and a 3D Convolution (3 x 3 x 3). The equation describing how the output feature map flow from layer l in ST-DenseNet's dense block is calculated as $fl = Hl([f0,f1,...,fl-1])$ where Hl is composition function (BN-ReLU-3DConv). The $[f0,f1,...,fl-1]$ is dense connectivity (concatenation operation) of the input feature maps from all the preceding layers. Each dense block is connected with the next one with a Transition Layer (TL). TL is comprised of two internal consecutive layers: 3D convolution and 3D pooling layers to resize the feature maps between dense blocks. Because of the Deep structure of ST-DenseNet, the problem of increasing number of input feature maps arises. The algorithms implemented to handle this problem namely, introduction of growth rate parameter and bottleneck layers are discussed in the next section. The total number of dense blocks used are 3 blocks which internally comprise of 4 layers each. The input of the ST-DenseNet is a resized cropped Bounding Box image with size of 100W x 100H of the pedestrians across 16 frames (≈ 0.5 sec of 30 FPS camera) from the Multiple Pedestrian Tracking stage. The output is two SoftMax classification probability scores (to cross or not) for each input sequence of the unique tracked pedestrians.

IV. ALGORITHMS USED

A. *Optimizer*

There are many algorithms which are used to train the deep networks at different stages of the architecture. The main algorithm used in both the networks is Adam as optimizer. The essence of the Adam optimizer can be squeezed into a single statement from the original paper which states as "The method computes individual adaptive learning rates for different parameters from estimates of first and second moments of the gradients; the name Adam is derived from adaptive moment estimation" [11]. Adam solves the problem constant learning rate which is predominant in SGD (Standard Gradient Descent) by adapting two key features from AdaGrad[12] and RMSProp[13]. Instead of adapting the parameter learning rates based on the average first moment (the mean) as in RMSProp, Adam also makes use of the average of the second moments of the gradients (the uncentered variance). ST-DenseNet model was trained using the Adam optimizer with a learning rate of 0.01 and batch size of 10 samples for 70 training epochs on NVIDIA Titan X GPU.

B. *Growth rate parameter*

Growth rate parameter is used to put a lid on increasing number of features in DenseNet architecture [9]. Each function Hl produces k feature maps, it follows that lth layer has k0+k x (l-1) input feature-maps, where k0 is the number of input layer. The k is referred as growth rate parameter. The growth rate regulates how much new information each layer contributes to the global state. Each layer adds k feature-maps of its own to this state.

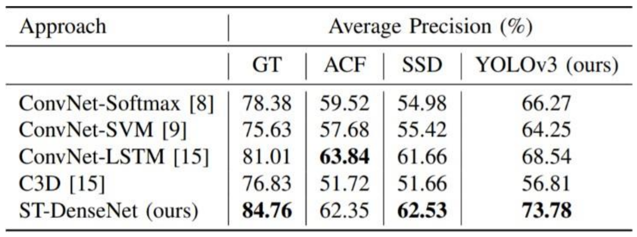
C. *Bottleneck Layers*

Bottleneck layers is also used as another approach to solve the problem. Although each layer only produces k output feature-maps, it typically has many more inputs. A 1×1x1 convolution is introduced as bottleneck layer before each 3×3 convolution to reduce the number of input feature-maps, and thus to improve computational efficiency.

V. RESULTS

A. *Average Precision*

Average Precision (AP) is the metric used to compare 4 baseline models as shown in Fig 4. The AP score is a summarization of the precision-recall curve in terms of a weighted average of precisions at different threshold values between 0.0 and 1.0. Total of 5 models were used to compare the results, ConvNet-SoftMax (based on Inception) [14], ConvNet-SVM (based on AlexNet)[5], ConvNet-LSTM(uses LSTM for time forecasting abilities)[15], C 3D (uses Conv2d to detect the object but takes input "a sequence of images" and uses LSTM to predict the case)[15]. The last one is proposed in the paper – combination of YOLOv3 and ST-DenseNet. Average precision score of 84.76% with real-time performance at 20 FPS is achieved.

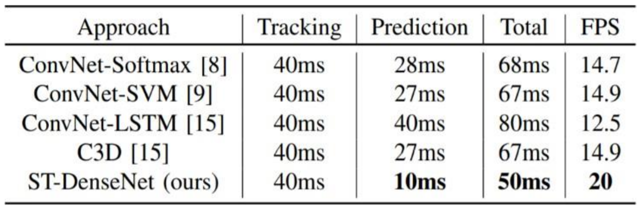


Approach	Average Precision (%)			
	GT	ACF	SSD	YOLOv3 (ours)
ConvNet-Softmax [8]	78.38	59.52	54.98	66.27
ConvNet-SVM [9]	75.63	57.68	55.42	64.25
ConvNet-LSTM [15]	81.01	**63.84**	61.66	68.54
C3D [15]	76.83	51.72	51.66	56.81
ST-DenseNet (ours)	**84.76**	62.35	**62.53**	**73.78**

Fig 4. Comparison of baseline models with proposed model based on *Average Precision* metric.

B. *Run Time Analysis*

The proposed model is compared with other baseline models for each stage as shown in Fig 5. It can be seen that r framework has achieved a real-time performance of 20 FPS, while the other baseline models were facing some challenges with the nearest model achieving only 14.9 FPS. All the run-time analysis experiments run on the same PC with an Intel i7 CPU and an Nvidia Titan X GPU.



Approach	Tracking	Prediction	Total	FPS
ConvNet-Softmax [8]	40ms	28ms	68ms	14.7
ConvNet-SVM [9]	40ms	27ms	67ms	14.9
ConvNet-LSTM [15]	40ms	40ms	80ms	12.5
C3D [15]	40ms	27ms	67ms	14.9
ST-DenseNet (ours)	40ms	**10ms**	**50ms**	**20**

Fig 5. Runtime Performance Analysis

every team member is required to complete this form, which will be reviewed and signed by other team members
by signing this form, you agree that the student's descriptions of contributions are factual
effort/contribution levels determine an individual's score which can be different from the rest of the team
This form does not contribute to page count.

Assignment name/key words Pedestrians Tracking using Spatio-Temporal DenseNet
Team #12

Student name:Rahavee Prabakaran	worked on literature	worked on implementation (data, platform, test run, debug, compatibility...)	generated results (run results, result data processing, presenting results	wrote report (Intro, method, result, discussions, ...)	other significant contributions	peer approval 1	peer approval 2	peer approval 3
specific & detailed evidence is required to support claims of contributions (make reference to specific paragrphs, equation #, figure #, code line #'s sections, etc...)	Found some papers related to the implementation of CNNs. Have put forward a promising paper regarding Satellite Imagery Road Extraction using CNNs. Worked with team members to scrutinize the availble literature.	NA	NA	* Worked on data preprocessing section for YOLOv3, SORT-UKF,ST-DenseNet. Involved in developing the final report.	*JAAD Dataset is analysed and it's statsics are explained clearly. *Summary of different CNN architecture using survey papers on Autonomous Driving Vehciles.	Udaya Gopi Kappala	Sanjay Mullapudi	Jaya Bhargavi Mannem

Student name:Sanjay Mullapudi	worked on literature	worked on implementation (data, platform, test run, debug, compatibility...)	generated results (run results, result data processing, presenting results	wrote report (Intro, method, result, discussions, ...)	other significant contributions	peer approval 1	peer approval 2	peer approval 3
specific & detailed evidence is required to support claims of contributions (make reference to specific paragrphs, equation #, figure #, code line #'s sections, etc...)	Found some papers related to the implementation of CNNs. Have put forward a promising paper regarding Pedestrian Tracking using CNNs. Worked with team members to scrutinize the availble literature.	NA	NA	*Wrote Network Architecture explantion. Multi object tracking Stage and ST-DenseNet are explained in great detail.	*DenseNet Architecture is compared with proposed Network. *Formatted the Final Version of the report	Jaya Bhargavi Mannem	Udaya Gopi Kappala	Rahavee Prabakaran

Student name:Jaya Bhargavi Mannem	worked on literature	worked on implementation (data, platform, test run, debug, compatibility...)	generated results (run results, result data processing, presenting results	wrote report (Intro, method, result, discussions, ...)	other significant contributions	peer approval 1	peer approval 2	peer approval 3
specific & detailed evidence is required to support claims of contributions (make reference to specific paragrphs, equation #, figure #, code line #'s sections, etc...)	Found some papers related to the implementation of CNNs. Have put forward a promising paper regarding Semantic Segmentation With Multi Scale Spatial Attention For Self Driving Cars using CNNs. Worked with team members to scrutinize the availble literature.	NA	NA	*Wrote Introduction to reference paper, Runtime Analysis in Results section. Collected Bibliography for the Report.	*Analysed different classsifiers used in baseline models for comparision in Performance Evaluation. *Organised team meetings.	Rahavee Prabakaran	Sanjay Mullapudi	Udaya Gopi Kappala

Student name:Udaya Gopi Kappala	worked on literature	worked on implementation (data, platform, test run, debug, compatibility...)	generated results (run results, result data processing, presenting results	wrote report (Intro, method, result, discussions, ...)	other significant contributions	peer approval 1	peer approval 2	peer approval 3
specific & detailed evidence is required to support claims of contributions (make reference to specific paragrphs, equation #, figure #, code line #'s sections, etc...)	Found some papers related to the implementation of CNNs. Have put forward a promising paper regarding Multimodal Trajectory Predictions for Autonomous Driving using CNNs. Worked with team members to scrutinize the availble literature.	NA	NA	*Wrote Algorthms used i.e., Adam Optimizer, growth rate paramter, Bottle Neck layers and Performance evaluation using Average Precision in Results.	*Initial Draft of report. *Discussed problem of increasing number of feature maps. *Deatailed explanation of Adam Optimizer.	Sanjay Mullapud	Jaya Bhargavi Mannem	Rahavee Prabakaran