2019 IEEE Conference on Information and Communication Technology (CICT)

Optimizing Driver Assistance Systems for Real-Time performance on Resource Constrained GPUs

Ojas A. Ramwala
Electronics Engineering
SVNIT
Surat, India
ojasramwala@gmail.com

Chirag N. Paunwala
E & C Engineering
SCET
Surat, India
chirag.paunwala@scet.ac.in

Mita C. Paunwala
E & C. Engineering
CKPCET
Surat, India
mita.paunwala@ckpcet.ac.in

Abstract—**The importance of Advanced Driver Assistance Systems has increased tremendously due to their ability to reduce road fatalities by facilitating drivers for appropriate action selection in circumstances involving high probability of collisions. One of the major factors contributing to accidents on road is driver distraction and drowsiness. A variety of algorithms including several Forward Collision Warning algorithms have been proposed to alleviate the issue to road accidents. These algorithms are promising approaches to mitigate this problem. However, most of these proposals are computationally complex algorithms and require powerful GPUs to perform in real-time. Such GPUs are not only expensive but also have high power consumption. Thus, it is necessary to yield real time performance on resource constrained GPUs like NVIDIA's Jetson TX2 which is not only one of the most eminent GPU-enabled platforms for autonomous systems but also cost effective and power efficient [1]. This paper proposes utilization of pruning of Neural Networks and TensorFlow TensorRT to optimize computationally complex algorithms utilized for Driver Assistance Systems to obtain real-time functionality on TX2 without compromising the accuracy of the system.**

Index Terms—Advanced Driver Assistance Systems, Forward Collision Warning, resource constrained GPUs, TensorFlow TensorRT

I. INTRODUCTION

Advanced Driver Assistance Systems (ADAS) are designed to enhance the capabilities of drivers for selection of suitable actions and to complement their perception while driving to minimize road fatalities by reducing human errors. Driver drowsiness and distraction are one of the major reasons for fatal road accidents. ADAS can be utilized to perform safety control, set off warnings, enhance perception or arouse attention to prospective hazards. Certain approaches suggest detection of driver's drowsiness or distraction and raise an alarm, while many algorithms have been developed to recognize possible risks of collision and take suitable actions to prevent such accidents. Detection of driver's inattentiveness may not always ensure correct judgement and can be unreliable because they depend on the physiological properties of the driver and it varies due to individual differences. Moreover, the decision making process of algorithms detecting potential accidents or driver's inappropriate behavior that may lead to risks must work in real-time to completely serve the purpose of assisting the processing units to perform inference in real-time. Such powerful processing units can accelerate inferencing, but they are very expensive. Thus, optimization of such algorithms for real time performance on resource constrained GPUs like NVIDIA's JETSON TX2 [1] is not only cost effective but also power efficient. This paper proposes methods to optimize approaches utilized for ADAS and implements it over AlexNet [2] for real-time functionality on resource constrained GPUs specifically for obstacle detection without reducing the accuracy of the algorithm.

II. RELATED WORK

A variety of approaches have been proposed to build efficient ADAS. A comparative study of relevant algorithms is done in this section.

Various ADAS utilize pedestrian detection systems to ensure safety of pedestrians. A variety of algorithms for pedestrian detection have been adopted. Approaches like AdaBoost cascading on Haar-like features [3, 4] and combining HOG with SVM [5, 6] were proposed for pedestrian detection. Although the approach suggested in [3] is fast, but it lacks robustness due its dependency on the appearance of pedestrians. On the other contrary, the approach based on the combination of HOG with SVM [5] is powerful but lacks speed.

Various ADAS address the issue of driver distraction. A variety of distraction indicators like sudden movements of the steering wheel, abrupt increase in the velocity, unusual increase in the distance from the leading vehicle, sluggish reactions to the reduction in speed of the leading vehicle, alteration in the driver's usual glance pattern or reduction in control and coordination of lateral movements of the driver have been considered in [7-11]. Since the detection of driver's distraction varies due to individual differences in the physiological properties, these systems can be unreliable.

Radar based systems for driver assistance have also been proposed in [12-15] for vehicle detection and blind spot identification. A laser-based system was also proposed in [16] which was limited only to collision avoidance and cannot be implemented for target driven navigation. Moreover, the systems based on lasers and radars are extremely expensive.

Approaches based on collision prevention like [17], utilize an image-based approach for feature extraction using imitation learning for obstacle avoidance. An algorithm based on Generative Adversarial Imitation Learning (GAIL) [18] for autonomous cars was proposed to learn driver models. In order to analyze and estimate the motion of obstacles, certain optical flow methods [19-21] were also proposed. The computational burden of such systems is very heavy and hence it is difficult to implement these methods on real-time embedded systems.

Stereo-vision based obstacle detection system has been proposed in [22], which directly obtains the depth information of the environment. However, the computational complexity of these stereo-vision based systems is huge and the cost is also significantly high.

Thus, numerous approaches have been explored in the field of ADAS. Several methods are efficient and have produced good results, but many of them suffer from high computational complexity and it becomes difficult to implement them on resource constrained GPUs. Various endeavors on Optimized Implementation of Deep Learning Models on the NVIDIA Jetson Platform have been undertaken. Certain hardware-level techniques [23] perform memory-level optimizations to GPU, such as using shared and texture memory, reducing accesses to global memory by using memory access coalescing and kernel fusion, etc. Another approach was to scale the frequency of CPU and/or GPU to trade-off performance with energy. Channel pruning is also an essential optimization technique. In order to accelerate Convolutional Neural Networks, dynamic channel pruning [24] was proposed with an idea to dynamically choose and then prune redundant features at the inference phase to reduce the computational cost. This method utilizes a "channel T-weighting" model to compute the importance of each channel for a specific layer of feature maps. If the corresponding weights of a channel are lower than then the given threshold, then such unimportant channels are removed. GPUs like NVIDIA JETSON TX2 are not only power efficient but also cost effective and it is important to optimize algorithms with high computational complexity for real-time performance on such GPUs.

III. Proposed Method

To explain the implementation of the optimization methods, this paper takes the example of the AlexNet [2] as the deep Convolutional Neural Network for the Forward Collision Warning algorithm. Fig. 1 shows the visualization of the AlexNet architecture with 5 convolutional layers and 3 fully connected layers.

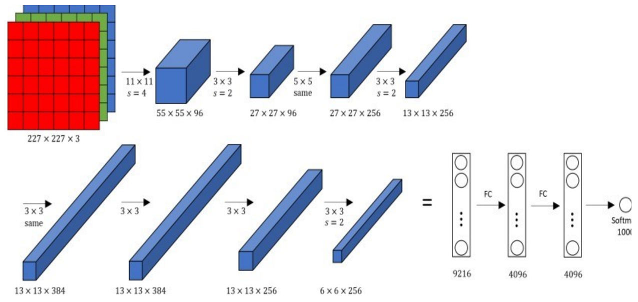

Fig. 1: AlexNet Architecture [2]

Since computational complexity is an important aspect, it is essential to mention that approximately 95% of the computation occurs at the convolutional layers and 5% occurs at the fully connected layers. The training phase of the algorithm involves feeding images from the camera and distance information from the GPS/IMU sensors. During the implementation, video from the camera is fed and frames are captured from the video using OpenCV. Every frame is saved as a matrix and processed by AlexNet [2] to predict whether it is "safe" or "danger" for the vehicle to move forward based on the score predicted by the application program interface. If the score is less than or equal to 0.5, the model predicts it as "danger" else predicts it as "safe". The algorithm infers the image and predicts the result correctly as shown in fig. 2. The algorithm was implemented on TensorFlow framework and typically took 15 seconds to infer a single image on NVIDIA Jetson TX2, which is obviously not real-time due to the high computational complexity of the algorithm.

In order to optimize this algorithm for real-time functionality, utilization of TensorFlow TensorRT [25] is proposed. TensorRT is a runtime optimizer which can be implemented for high-performance inferencing in lower precisions like floating point 16 (FP16). Deep learning models can be implemented over lower precisions to reduce inference time and minimize the energy spent in computing without practically any compromise on accuracy. FP16 reduces the memory consumption by lowering the size of the tensors to half and also decreases the training time by minimizing the arithmetic and network bandwidth and hence speeding up computations on GPU.

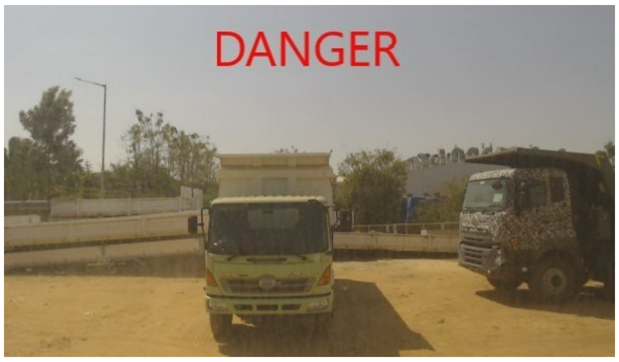

Fig. 2: Prediction of the proposed algorithm

A variety of important transformations are done on the neural network graph. To prevent unnecessary computations, layers with unused output are eliminated by TensorRT. Moreover, aggregation of horizontal layers and fusion of convolutional, bias and ReLU layers is performed to reduce the burden of heavy computation. In order to utilize the TensorFlow TensorRT optimization, it is necessary to convert the protocol buffer file obtained after training the neural network model to its corresponding TensorFlow TensorRT engine. TensorRT then attempts to find the fastest implementation of the particular model, by strengthening a varied agglomeration of highly optimized kernels. Fig. 3 provides the visualization of the architecture obtained after the transformations performed by TensorFlow TensorRT. TensorFlow TensorRT substantially reduces the size of the architecture by introducing two operators in the architecture. These operators replace various convolutional, bias and ReLU layers with an equivalent function of lower time complexity. TensorFlow TensorRT thus produces significant improvement in results and can be utilized if such a relative optimization in results is required.

Another optimization step proposed is the Pruning of the neural network. Pruning neural networks is an important step to eliminate redundant parameters and thus reduce computations. Pruning is an iterative process to fine-tune a model to reduce the computations to a minimum acceptable level without impermissible compromise in accuracy. Pruning should be performed till real-time results for inferencing are obtained as shown in Fig. 4. Pruning can be performed for a smaller model and for improved speed. Model size can be significantly reduced by pruning fully connected layers but that can lead to drastic reduction in the accuracy. This paper focuses on removing multiple channels from convolutional layers in order to minimize the computations. Experimental approach is proposed to perform pruning efficiently to record the inference time and accuracy of the model. This paper proposes the implementation of channel pruning technique for Keras models suggested in [26]. In this technique, L1-norm of each filter's weight is calculated, and removes the ones with least L1-norm. In the experimental approach, for the AlexNet [2] architecture implemented in Keras framework, channels are removed from each convolution layer till there is significant reduction in inference time without any major reduction in accuracy. This approach is then extended for combination of multiple layers and performed iteratively till real time performance is obtained.

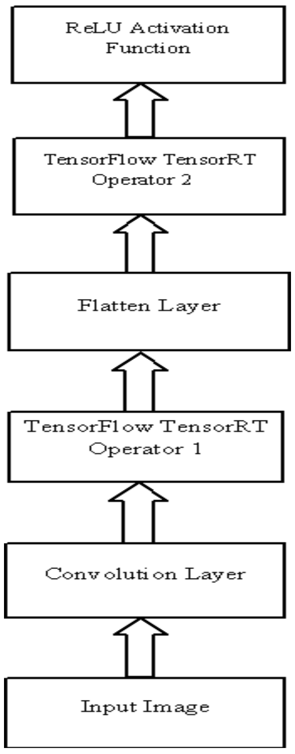

Fig. 3: Optimized model architecture [FP16]

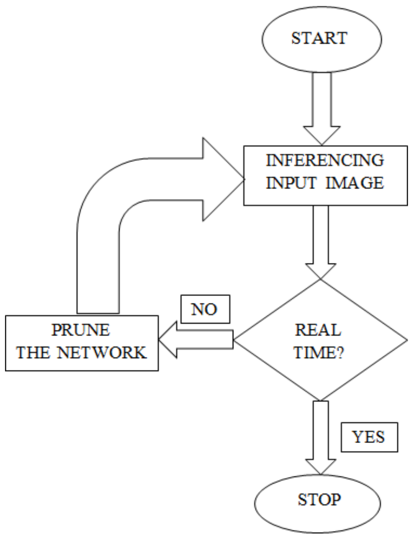

Fig. 4: Pruning for real-time inferencing

IV. Experiments and Results

Accuracy of the original architecture with approximately 28 million (28,038,481) parameters is depicted in the confusion matrix in Table 1. The original model has accuracy of 78.5% and an inference time of approximately 15 seconds for a single image on TX2 as presented in Table 2.

TABLE I. CONFUSION MATRIX OF ORIGINAL ARCHITECTURE

	Predicted: Safe	*Predicted: Danger*
Actual : Safe	696 (TP)	221 (FP)
Actual: Danger	0 (FN)	123 (TN)

Here TP, FP, FN and TN denote True Positive, False Positive, False Negative and True Negative respectively

TABLE II. PERFORMANCE OF ORIGINAL MODEL

Accuracy	*Inference Time*
78.5%	15 seconds

Due to optimizations performed by TensorFlow TensorRT (TF-TRT), the model now yields an inference time of 7 seconds on TX2, which is over 50% improvement with the same accuracy of 78.75% as presented in Table 3

TABLE III. PERFORMANCE OF TF-TRT OPTIMIZED MODEL

Accuracy	*Inference Time*
78.5%	7 seconds

Fig 5. Shows the results of pruning multiple channels from individual convolution layers and the corresponding inference time. The vertical axis represents the number of channels pruned and the horizontal axis represents the inference time.

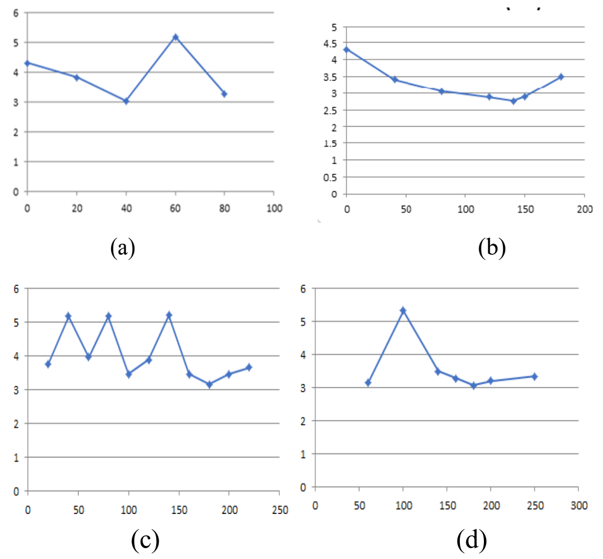

Fig. 5: Pruning for Convolution Layer (a) 1 (b) 2 (c) 3 (d)4

In order to achieve a trade-off between accuracy and speed, suitable combination of multiple channels pruned from multiple convolutional layers must be chosen for least inference time and maximum acceptable accuracy. From analysis of the graphs shown in Fig. 5, the combination presented in Table 4 is selected.

TABLE IV. DETAILS OF NEURAL NETWORK PRUNING

Convolution Layer	*CL 1*	*CL 2*	*CL 3*	*CL 4*
Number of channels	96	384	384	256
Channels Pruned	40	140	180	180

Accuracy of the pruned model with approximately 23.8 million (23,796,701) parameters is depicted in the confusion matrix in Table 5.

TABLE V. CONFUSION MATRIX OF PRUNED ARCHITECTURE

	Predicted: Safe	*Predicted: Danger*
Actual : Safe	666 (TP)	251 (FP)
Actual: Danger	0 (FN)	123 (TN)

The pruned architecture has an accuracy of 75.86% with an inference time of 0.01 second on TX2 as presented in Table 6.

TABLE VI. PERFORMANCE OF THE PRUNNED MODEL

Accuracy	*Inference Time*
75.86%	1 msec

Thus, producing real-time results on a resource constrained GPU.

V. CONCLUSION

A Forward Collision Warning framework dedicated to notify the driver regarding potential risks of any road accident has been considered to illustrate the approaches of optimizing computationally complex algorithms for real-time performance on the resource constrained Jetson TX2 GPU which is one of the most eminent GPU-enabled platforms for autonomous systems. Implementation of the TensorFlow TensorRT runtime optimizer has been proposed for high-performance inferencing over lower precision. In order to remove redundant parameters, neural network pruning has been proposed to lower the computations and hence the inference time. Real- time performance over a resource constrained GPU for Driver Assistance Systems is thus obtained without any significant reduction in accuracy.

REFERENCES

[1] https://developer.nvidia.com/embedded/jetson-tx2

[2] Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton in NIPS'12 Proc. of the 25th International Conference on Neural Information Proceedings Systems – Volume 1 pages 1097 – 1105, 2012.

[3] P. Viola, M. J. Jones, and D. Snow, "Detecting pedestrians using patterns of motion and appearance," Int. J. Comput. Vis., vol. 63, no. 2, pp. 153–161, Jul. 2005.

[4] D. Gerónimo, A. D. Sappa, A. López, and D. Ponsa, Adaptive Image Sampling and Windows Classification for On-board Pedestrian Detection. Bielefeld, Germany: Univ. Bielefeld, 2007.

[5] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in Proc. IEEE Comput. Soc. Conf. CVPR, vol. 1, pp. 886–893, 2005.

[6] L.Zhang, B.Wu, and R.Nevatia, "Pedestrian detection in infrared images based on local shape features," in Proc. IEEE Conf. CVPR, pp. 1–8, June 2007.

[7] W. J. Horrey and D. J. Simons, "Examining cognitive interference and adaptive safety behaviours in tactical vehicle control," Ergonomics, vol. 50, no. 8, pp. 1340–1350, 2007.

[8] Y.-C. Liu, "Comparative study of the effects of auditory, visual and multimodality displays on drivers' performance in advanced traveller information systems," Ergonomics, vol. 44, no. 4, pp. 425–442, 2001.

[9] B. Donmez, L. N. Boyle, and J. D. Lee, "Safety implications of providing real-time feedback to distracted drivers," Accident Anal. Prevention, vol. 39, no. 3, pp. 581–590, 2007.

[10] Y. Liang, "Detecting driver distraction," Ph.D. dissertation, Dept. Ind. Eng., Graduate College Univ. Iowa, Iowa City, IA, USA, 2009.

[11] L. S. Angell et al., "Driver workload metrics task 2 final report," Nat. Highway Traffic Safety Admin., Washington, DC, USA, Tech. Rep. DOT HS 810 635, 2006.

[12] Bogomil Shtarkalev, Bernard Mulgrew, Multistatic moving target detection in unknown coloured Gaussian interference, Signal Processing. 115 pp.130-143, 2015.

[13] Bo Du, L.P. Zhang, Target detection based on a dynamic subspace, Pattern Recognition. 47, pp. 344-358,2014.

[14] X.R. Chen, C.L. Xi, J.H. Cao, Research on moving object detection based on improved mixture Gaussian model, Optik, pp. 1-4, 2015.

[15] Hugh L. Kennedy, Multidimensional digital smoothing filters for target detection, Signal Processing. 114, pp. 251-264, 2015.

[16] J. Sergeant, N. S¨underhauf, M. Milford, and B. Upcroft, "Multimodal deep auto encoders for control of a mobile robot," in Proc. Australas. Conf. Robot. Automat. pp. 1–10, 2015

[17] U.Muller, J. Ben, E.Cosatto, B. Flepp, and Y. L. Cun, "Off-road obstacle avoidance through end-to-end learning" in Proceedings of Adv. Neural Inf. Process. Syst., pp. 739–746, 2005.

[18] AJ. Ho and S. Ermon, "Generative adversarial imitation learning," in Proc. Adv. Neural Inf. Process. Syst., pp. 4565–4573, 2016

[19] S. Denman, V. Chandran, and S. Sridharan, "An adaptive optical flow technique for person tracking systems," Pattern Recognition Letter, vo.28, pp. 1232-1239, 2007.

[20] J. D. Alonso, E. R. Vidal, A. Rotter. and M. Muhlenberg, "Lane-change decision aid system based on motion-driven vehicle tracking," IEEE Transactions on Vehicular Technology, vol. 57, no5, pp. 2736-2746, 2008.

[21] P. Lenz, J. Ziegler, A. Geiger, and M. Roser, "Sparse scene flow segmentation for moving object detection in urban environments," Proceeding of the IEEE Intelligent Vehicles Symposium, pp. 926-932, June. 2011.

[22] https://docs.nvidia.com/deeplearning/frameworks/tf-trt-user- guide/index.html.

[23] Mittal, Sparsh. A Survey on Optimized Implementation of Deep Learning Models on the NVIDIA Jetson Platform, Journal of Systems Architecture, January 2019.

[24] Chiliang, Zhang & Hu, Tao & Yingda, Guan & Zuochang, Ye. (2019). Accelerating Convolutional Neural Networks with Dynamic Channel Pruning.

[25] M. Bertozzi, A. Broggi, and A. Facioli, "Stereo inverse perspective mapping: theory and applications," Image and Vision Computing, vol. 16, pp. 585-590, 1998.].

[26] https://github.com/BenWhetton/keras-surgeon