Making Lane Detection Efficient for Autonomous Model Cars

Anthony B Song[1], Riley Francis[1], Kanishk Tihaiya[1], Jiangwei Wang[2], Shanglin Zhou[2], Fei Miao[2], Caiwen Ding[2]
[1]*Edwin O Smith High School, Storrs, CT,* [2]*University of Connecticut, Storrs, CT*
Contact email: songa23@eosmith.org

***Abstract*—In recent years, autonomous driving has attracted significant attention. There are many technical challenges in creating such automation, such as quick recognition of objects in the environment. A lane-detection system is a crucial component of connected and autonomous vehicles. Although deep learning models have been the state-of-the-art for lane detection from camera footage collected by vehicle sensors, these models require extensive computation and memory storage to detect and track lanes, which restricts their applicability. We apply a model compression technique to prune weights for a well-trained ResNet-18 lane detector. The compressed model can run on a power-efficient TX2 or FPGA equipping an F1Tenth model car. We will demonstrate this system using our assembled model car and report detection accuracy and efficiency using benchmark datasets.**

I. Project Description

A. Model Car Assembly

The F1Tenth car is an inexpensive test bed for validating autonomous driving algorithms. The car was constructed using a racing car chassis, motors, and sensors, which are linked together to an NVIDIA Jetson TX2 low power module. Our design uses the TX2 because it is suitable for small car chassis and has a GPU that allows for the deployment of CNN based object and lane detection. A VESC Module controls the motors and a customized power board is connected to a lithium ion battery to power the car. We hope this design achieves both low power and fast processing.

The Robot Operating System (ROS) enables the car to make decisions based upon data from the Hokuyo LIDAR and a visual camera. ROS uses topics and nodes to link robotic components together through processes called publishing and subscribing.

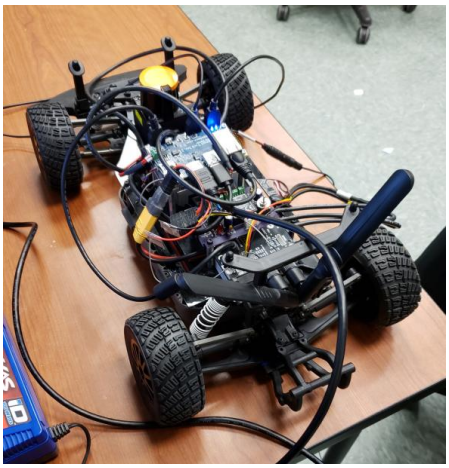

Fig. 1. A fully assembled vehicle.

B. Lane Detection

Lane detection has a fundamental problem in that lane markings are the main static component on the road that instruct the vehicles to interactively and safely drive on the way. Deep learning methods that are applied to lane detection usually treat the problem as a semantic segmentation task [1]. VPGNet [2] proposes a multi-task network guided by vanishing points for lane and road marking detection. It tries to address inaccurate detection under poor weather conditions. SCNN [3] tries to use visual information more efficiently by aggregating information from different dimensions via processing sliced features and adding them together one by one. Other methods focus on real-time running of the algorithm, such as SAD [4] that applies attention distillation mechanism to improve the representation learning of CNN-based lane detection models.

The lane detection model deals with camera images and detects a precise coordinate for each lane markings. Method were applied from Ultra-Fast-Lane-Detection [5]. Ultra-Fast-Lane-Detection uses ResNet-18 as a backbone for global context, aggregating auxiliary segmentation tasks that utilizes multi-scale features by extracting middle step feature maps to model local features. This method is not only light and effective for embedded computing device, but also addresses the no-visual-clue problem in the lane detection area. Meaning if the lane markings are blurred, affected by light, or even completely obscured, this Ultra-Fast-Lane-Detection method can still accurately detect the lane markings.

Instead of segmenting every pixel of each lane, images are decomposed to a collection of rows called row anchors. Lane detection is redefined as finding a set containing positions of lane markings in certain rows of the image, i.e., row-based selection and classification based on global image. As shown in Fig. 2, if images have size $H \times W$, usually h row anchors are selected. Each row anchor is then divided into many grids/cells. The location of lanes can be converted to localize certain cells over these row anchors. Comparing with traditional segmentation approaches that need to deal with $H \times W$ classification problems, Ultra-Fast-Lane-Detection only needs to deal with classification problems on h rows. The only exception being that the classification on each row is W-dimensional. It simplifies the original $H \times W$ classification problems to only h classification problems. Generally, h is much smaller than the image height H.

To handle challenging scenarios such as severe occlusion and extreme lighting conditions, Ultra-Fast-Lane-Detection uses whole images as a receptive field to maintain global features. Contextual information from other parts of the image can be utilized. Auxiliary segmentation learns from prior layers' feature maps, so information like shape and direction of lanes can be learned and leveraged in the main classification task.

Evaluation metrics for lane detection tasks under individual data sets are distinctly discrete. Accuracy is sometimes used as a percentage of the correctly predicted number of lane

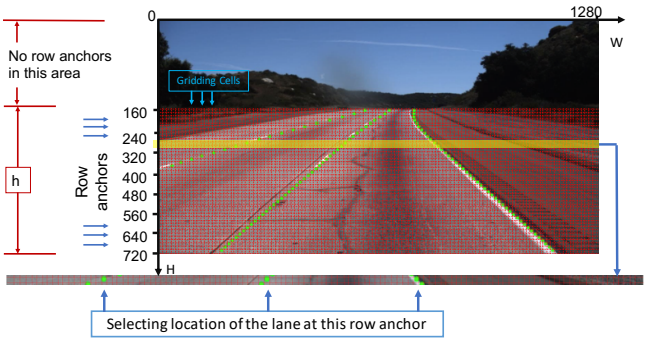

Fig. 2. Illustration of row anchors and decomposition of lanes. Row anchors are predefined as horizontally selecting rows in the images. Each lane marking is decomposed into grid cells in row anchors.

points [6]. $F1 = \frac{2\times Precision\times Recall}{Precision+Recall}$ is another criteria [3].

C. Weight Pruning to Speed Up Processing

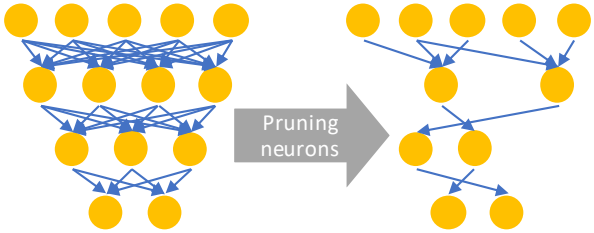

Fig. 3. Illustration of weight pruning for DNNs. Left: network before pruning, right: network after pruning, removing several neurons and synapses [7].

Many investigations have shown that there exists redundancy in DNN model parameters [8]–[10]. As shown in Fig. 3, effective model compression with negligible accuracy loss can be achieved using weight pruning methods. One fundamental work is [8], that uses a three-step method prunes redundant connections in DNNs. This work reduces $9\times$ number of parameters in AlexNet on the ImageNet dataset without accuracy degradation. However, indices are needed to locate which weight to pruning. This is not friendly in hardware implementations for low-performance improvement [11]. This problem is partially addressed by several works [12], [13]. Energy efficiency-aware pruning method [14], [15] facilitates energy-efficient hardware implementations [16], allowing for certain accuracy degradation, and structured sparsity learning technique is proposed for irregular network structure after pruning.

Processing time of 3D object detection using PointPillars is significantly longer ($11.16\times$) than lane detection using Ultra-Fast-Lane-Detection. To synchronously obtain the neighbouring vehicles' location and lane markings' information, we adopt Alternating Direction Method of Multipliers (ADMM)-based weight pruning technique [17], [18] on PointPillars network to reduce its running time.

If we consider a general N-layer DNN with loss function $f\left(\{\mathbf{\Theta}_i\}_{i=1}^N\right)$, the overall problem of DNN weight pruning is minimizing this loss function subject to i^{th} layer's weight belongs to $\{\mathbf{\Theta}_i \mid \text{card}\left(\text{supp}\left(\mathbf{\Theta}_i\right)\right) \leq t_i\}$, where t_i is desired numbers of non-zero weights. We can then use indicator function and incorporate auxiliary variables to re-formulate the loss function to Eq. 1 as

$$\begin{aligned} \underset{\{\mathbf{\Theta}_i\}}{\text{minimize}} \quad & f\left(\{\mathbf{\Theta}_i\}_{i=1}^N\right) + \textstyle\sum_{i=1}^N g_i\left(\mathbf{P}_i\right) \\ \text{subject to} \quad & \mathbf{\Theta}_i = \mathbf{P}_i, i = 1, \ldots, N \end{aligned} \tag{1}$$

Eq. 1 can then be decomposed into two sub-problems through application of the augmented Lagrangian [18], and can be solved iteratively until convergence.

Euclidean projection is performed during training to guarantee most weights in each layer are non-zero. Then zero-weights are masked and the network is retrained until converge.

D. Evaluation Metrics

We will study the performance of our solutions (i) from the application perspective, i.e., images/s, and (ii) from the system point of view with respect to latency (s), throughput (GFLOPS), and energy-efficiency/power (W). An environment will be setup for the F1TENTH cars that have implemented ultra fast lane detection. We also want to measure the lane detection accuracy. We will explore the trade-off of accuracy and energy efficiency.

References

[1] J. Kim and M. Lee, "Robust lane detection based on convolutional neural network and random sample consensus," in *International conference on neural information processing*. Springer, 2014.

[2] S. Lee, J. Kim, J. Shin Yoon, S. Shin, O. Bailo, N. Kim, T.-H. Lee, H. Seok Hong, S.-H. Han, and I. So Kweon, "Vpgnet: Vanishing point guided network for lane and road marking detection and recognition," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 1947–1955.

[3] X. Pan, J. Shi, P. Luo, X. Wang, and X. Tang, "Spatial as deep: Spatial cnn for traffic scene understanding," *arXiv preprint arXiv:1712.06080*, 2017.

[4] Y. Hou, Z. Ma, C. Liu, and C. C. Loy, "Learning lightweight lane detection cnns by self attention distillation," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 1013–1021.

[5] Z. Qin, H. Wang, and X. Li, "Ultra fast structure-aware deep lane detection," *arXiv preprint arXiv:2004.11757*, 2020.

[6] TuSimple, "Tusimple competitions for cvpr2017," https://github.com/TuSimple/tusimple-benchmark, 2020.

[7] A. Ren and et al., "Admm-nn: An algorithm-hardware co-design framework of dnns using alternating direction methods of multipliers," in *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*, 2019, pp. 925–938.

[8] S. Han, J. Pool, J. Tran, and W. Dally, "Learning both weights and connections for efficient neural network," in *NeurIPS*, 2015.

[9] J.-H. Luo, J. Wu, and W. Lin, "Thinet: A filter level pruning method for deep neural network compression," in *Proceedings of the IEEE international conference on computer vision*, 2017.

[10] Y. Wang, C. Wang, Z. Wang, and et al., "Mcmia: Model compression against membership inference attack in deep neural networks," *arXiv preprint arXiv:2008.13578*, 2020.

[11] B. Li, Z. Kong, T. Zhang, J. Li, Z. Li, H. Liu, and C. Ding, "Efficient transformer-based large scale language representations using hardware-friendly block structured pruning," *arXiv preprint arXiv:2009.08065*, 2020.

[12] T. Zhang, X. Ma, Z. Zhan, and et al., "A unified dnn weight compression framework using reweighted optimization methods," *arXiv preprint arXiv:2004.05531*, 2020.

[13] Y. Gong, Z. Zhan, Z. Li, and et al., "A privacy-preserving-oriented dnn pruning and mobile acceleration framework," in *Proceedings of the 2020 on Great Lakes Symposium on VLSI*, 2020, pp. 119–124.

[14] T.-J. Yang, Y.-H. Chen, and V. Sze, "Designing energy-efficient convolutional neural networks using energy-aware pruning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5687–5695.

[15] D. Gurevin, S. Zhou, L. Pepin, B. Li, M. Bragin, C. Ding, and F. Miao, "A surrogate lagrangian relaxation-based model compression for deep neural networks," *arXiv preprint arXiv:2012.10079*, 2020.

[16] B. Li, S. Pandey, H. Fang, Y. Lyv, J. Li, J. Chen, M. Xie, L. Wan, H. Liu, and C. Ding, "Ftrans: energy-efficient acceleration of transformers using fpga," in *Proceedings of the ACM/IEEE International Symposium on Low Power Electronics and Design*, 2020, pp. 175–180.

[17] T. Zhang and et al., "A systematic dnn weight pruning framework using alternating direction method of multipliers," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[18] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein *et al.*, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends® in Machine learning*, 2011.