ADAPTIVELY WEIGHTED MULTI-TASK LEARNING USING INVERSE VALIDATION LOSS

Waseem Abbas[⋆†] *and Murtaza Taj*[†]

[⋆] Cloud Application Solutions Division, Mentor, A Siemens Business, Lahore, Pakistan
[†] Department of Computer Science, Syed Babar Ali School of Science and Engineering
Lahore University of Management Sciences (LUMS), Lahore, Pakistan

muhammad_waseem@mentor.com, murtaza.taj@lums.edu.pk

ABSTRACT

Multi-task learning aims to enhance the performance of a model by inductive transfer of information among tasks. However, joint optimization of multiple tasks is challenging due to unbalanced data ranges and variations in task difficulties which can cause the model to converge only for a single task which has large values. To address these problems, we propose a novel weighting scheme based on validation loss. The proposed weighted scheme is evaluated on three datasets, including publicly available Comma.ai and Udacity benchmark dataset and GTA-V dataset. Our experiments demonstrate the superior performance of the proposed approach compared to the current state-of-the-art methods.

Index Terms— Deep Learning, Adaptive Task Weights, Convolutional Neural Network (CNN), Autonomous Driving

1. INTRODUCTION

A human can learn multiple tasks at a time and multitasking capability can be achieved by prioritizing difficult tasks over easy tasks [1]. Like humans, intelligent machines or models can also learn multi-tasks by training on multiple tasks simultaneously and it is frequently used in several applications, including natural language processing [2], computer vision [3] and speech signal processing [4]. However, when model trained in such a manner to perform multi-tasks, easy tasks can dominate learning by ignoring harder tasks. This posed an important question: what is the right balance between learning of multiple tasks from easy versus hard tasks? To address the problem a diverse domain, multi-tasks learning can be categorized into two approaches: i) task weighting or ii) sharing information between the linked tasks.

In task weighting scheme, the objective function is obtained by assigning weights, automatically or via external human supervision, to each task according to their difficulty level and task weights [5]. Consideration of task difficulty and uncertainty is a major challenge in this approach [6]. In [7], the authors proposed an altering scheme to switch between task ordering and instance ordering and trained task-specific model in a single iteration without gradient descent.

Information sharing schemes among the tasks share common features but maintain separate task-specific modules. This can be categorized into hard parameter sharing, soft parameter sharing and task-hierarchy. In parameter sharing approaches, [8] is a seminal work, able to predict segmentation, pose estimation and human detection. In [9], researcher proposed a hierarchical approach by supervising easier tasks at lower layers and harder tasks at later layers. A critical assumption in multi-tasks learning is that the underlying distribution across all the tasks remains the same. However, this assumption is broken when defining a multi-task problem over dynamic tasks. On the other hand, parameter sharing can lead to a critical layer responsible for learning representations of discriminative downstream tasks. Further, the selection of the representation layer is difficult for a dynamic multi-task problem.

Autonomous vehicle control has grasped substantial research and commercial attention in the last couple of years. It requires effective digital control of steering, acceleration and braking systems for safe driving. Estimation of these parameters requires the understanding of the surrounding environment, under various lighting and weather conditions, and dynamic range of tasks makes it challenging. Many proposed methods estimate a single parameter such as only steering angle from one or more images [10, 11]. In [12], researchers proposed three end-to-end deep learning architectures including CNN with a fully convolutional network (CNN+FCN), CNN with attention mechanism (CNN+Attention) and CNN with LSTM cells (CNN+LSTM). In [13], the researcher explored transfer learning, 3D LSTM, and baseline model (Predict zero) on Udacity [14] dataset for estimation of steering angle. In [15], an end-to-end deep learning model using LSTM and CNN+LSTM architectures was proposed. similarly, [12] introduced attention mechanism and proposed three end-to-end deep learning architectures including CNN+FCN, CNN+Attention, and CNN+LSTM. These techniques addressed single task learning problem, however, learning of multiple tasks using a single model is a challenging and unexplored area.

MultiNet [16] combined classification, detection and segmentation in a single architecture. It is based on ResNet [17] and consists of three shared encoding layers followed by task independent decoding layers. They used a softmax cross-entropy for segmentation and classification and a sum of absolute differences for four values of the bounding box prediction. The different losses were naively summed together to achieve the combined loss.

In this paper, we propose a solution for task weighting scheme to predict multiple outputs present in a scenario. In addition to the weighting scheme, we propose a multi-task architecture with soft weight sharing technique by observing shared information among the tasks. Our scheme is encouraged to prioritize harder tasks over easier tasks. Empirically, we evaluate the proposed method on regression problem to predict steering angle, acceleration and brake simultaneously using the publicly available Comma.ai [18], Udacity [14] datasets. Further, we generated synthetic dataset in a gaming environment (GTA-V) and tested in a live gaming environment.

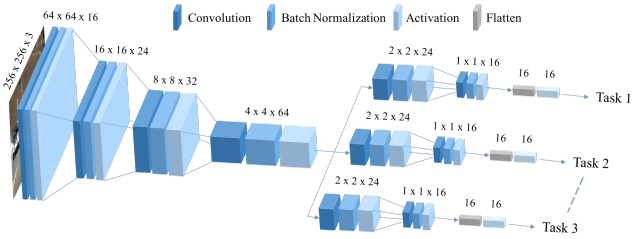

Fig. 1. Proposed Model for the MTL problem.

2. PROPOSED METHOD

We propose an adaptive weighting scheme in addition to a CNN based deep architecture containing shared layers as well as task independent layers for Multi-Task Learning (MTL). The proposed scheme assign more weights to the harder tasks and use previous validation loss to determine relative task difficulties. To evaluate the proposed model and task weighting scheme, we apply MTL to autonomous driving and show that the task weighting scheme influence the performance of an individual task by prioritizing the tasks which require more training iterations to be learned.

2.1. Weighting Scheme

Training a model for multitasking problem with unbalanced data ranges can converge a model for only a single class which has large values. In such cases, optimizer tries to minimize the largest error in loss function and converges in that direction which decreases the significance of other parameters.

2.1.1. Objective Function

Let $\mathbf{x} = \{x_i | i = 1, 2, ..., N\}$ is the set of input images from training samples and $\mathbf{y}_{D_i} = \{y_{D_i} | i = 1, 2, ..., N\}$ is the set of corresponding labels of $\mathbf{x}$. Where N is the total number of training samples. For the ease of presentation of network parameters, we define $\mathbf{W} = \{\mathbf{W}_1, \mathbf{W}_2, ..., \mathbf{W}_L\}$ and $\mathbf{B} = \{\mathbf{B}_1, \mathbf{B}_2, ..., \mathbf{B}_L\}$, where $\mathbf{W}_l$ and $\mathbf{B}_l$ are the weights and biases. Let ρ denotes weights ($\mathbf{W}$) and bias ($\mathbf{B}$) as combined parameters. In order to learn all parameters, apart from minimizing the combined loss, we propose objective function as $\mathcal{L}_D = min\Big(\sum_j \omega_{\theta^j} \mathcal{L}_{\theta^j}(x, \rho)\Big)$, where ω_{θ^j} is the weighting factor that defines the contribution of the j^{th} individual task and $\mathcal{L}_{\theta^j}$ is the individual loss function for the j^{th} task. We used Mean Squared Error (MSE) as Loss function/Objective function and it can be formulated as:

$$\mathcal{L}_D = min\Big(\sum_j \omega_{\theta^j} \frac{1}{\beta} \sum_{i=1}^{\beta} (\theta_i^j - \hat{\theta}_i^j)^2\Big) \tag{1}$$

For the estimation of multiple tasks ($\hat{\mathbf{y}}_\mathbf{D}$), we trained the proposed model with three different configurations of class weights.

- Scheme I: In the first configuration, the proposed model is trained for all tasks with an equal weighting scheme (ω_{θ^j}). This approach can over-fit the training for a task which has large values as the goal of the objective function is to minimize the cumulative loss of three tasks.

Table 1. Summary of state-of-the-art autonomous driving datasets used for evaluation of the proposed method.

	Comma.ai	**Udacity**	**GTA-V**
Frames	522,434	39,422	700,000
FPS	20Hz	20 Hz	20 Hz
Hours	7.25 hrs	8 hrs	194 Hz
Condition	Highway/Urban	Urban	Highway/Urban
Location	CA/USA	CA/USA	CA/USA
Lighting	Day/Night	Day	Day

- Scheme II: In the second configuration, we assign adaptive weighting factors to individual task by obtaining the initial weights from the first configuration. Let $\mathcal{L}_\theta^{j,v}$ is the final validation loss obtained from the first configuration. Using these validation losses, weighting factors for objective function are computed as $\omega_{\theta^j} = \frac{1}{\mathcal{L}_\theta^{j,v}}$. This configuration assigns an equal loss to each individual task and minimization of combined loss tries to converge the model for each equally. This scheme introduces task specific weighting. Minimization of combined loss tries to converge the model for tasks which were not better learned during Scheme 1.
- Scheme III: In third configuration, hard weighting scheme is applied as $\omega_s = 0.1$, $\omega_a = 0.2$ and $\omega_b = 0.7$ for driving control tasks. These weighting factors are found empirically by observing the range and statistics of the datasets.

The proposed model is trained for all three configurations and influence of weighting factor is analyzed later in Section 3.1 which shows that the adaptive weighting scheme (scheme II) supersedes all its alternatives.

2.1.2. Optimization and Weights Update

We employed commonly used Stochastic Gradient Descent (SGD) method to obtain the network parameters, $\mathbf{W}$ and $\mathbf{B}$, of the proposed model. Let the model has $L + 1$ layers where $(L + 1)$-th layer denotes the regression layer and l-th layer is an intermediate layer. Output of regression layer $\mathbf{h}_{L+1}(x_i)$, feature layer $\mathbf{h}_L(x_i)$ and any intermediate layer $\mathbf{h}_l(x_i)$ can be computed as:

$$\mathbf{h}_{L+1}(x_i) = \Psi_{L+1}(\mathbf{W}_{L+1}\mathbf{h}_L(x_i)) + \mathbf{B}_{L+1} \tag{2}$$

$$\mathbf{h}_L(x_i) = \frac{\Psi_L(\mathbf{W}_L\mathbf{h}_{L-1}(x_i)) + \mathbf{B}_L}{||\Psi_L(\mathbf{W}_L\mathbf{h}_{L-1}(x_i)) + \mathbf{B}_L||_2} \tag{3}$$

$$\mathbf{h}_l(x_i) = \Psi_l(\mathbf{W}_l\mathbf{h}_{l-1}(x_i)) + \mathbf{B}_l \tag{4}$$

where x_i is given input, Ψ_{L+1} is the regression function and Ψ_l is an activation function.

Gradients of objective function $\mathcal{L}_D$ with respect to weights $(\mathbf{W}_{L+1}, \mathbf{W}_L)$ and $(\mathbf{W}_L, \mathbf{B}_L)$ can be computed as:

$$\frac{\partial \mathcal{L}_D}{\partial \mathbf{W}_l} = \frac{\partial}{\partial \mathbf{W}_l} \sum_j \omega_{\theta^j} \frac{2}{\beta} \sum_{i=1}^{N} \nabla_l \mathbf{h}_{L-1}^{\theta^j}(x_i)^T \tag{5}$$

$$\frac{\partial \mathcal{L}_D}{\partial \mathbf{B}_l} = \frac{2}{\beta}\Big(\sum_j \omega_{\theta^j} \sum_{i=1}^{\beta} \nabla_l\Big) \tag{6}$$

where ∇ is the gradient. SGD computes gradients for multiple samples, mini-batch size β, instead of computing the gradients for each step. Updated learning parameters can be computed as:

$$\nabla_{L+1} = \mathbf{h}_{L+1}(x_i) - y_i \tag{7}$$

Table 2. Comparison of validation loss (MSE) of all three weighted loss schemes.

Dataset	Scheme	Steering angle	Acceleration	Brake	Torque	Speed
Comma.ai	I	310	0.024	747	-	-
	II	**304**	**0.011**	**710**	-	-
	III	324	0.022	736	-	-
Udacity	I	0.004	-	-	0.021	0.419
	II	**0.002**	-	-	**0.015**	**0.395**
	III	0.004	-	-	0.017	0.412
GTA-V	I	1.698	-	-	1.206	0
	II	**1.238**	-	-	**1.115**	0
	III	1.724	-	-	1.192	0

Table 3. Evaluation of estimation of steering angle in terms of MAE and SD on Comma.ai dataset [18].

MODEL	MAE	SD
CNN+FCN [12]	2.54	3.19
CNN+LSTM [12]	2.58	3.44
CNN+ATTENTION($\lambda = 0$) [12]	2.52	3.205
CNN+ATTENTION ($\lambda = 10$) [12]	2.56	3.51
CNN+ATTENTION ($\lambda = 20$) [12]	2.44	3.20
CNN [19]	2.42	3.26
PROPOSED	**1.106**	**1.65**

$$\nabla_L = \mathbf{W}_{L+1}^T \nabla_{L+1} \odot \mathbf{Q}(\mathbf{O}_L(x_i)) \Psi'(\mathbf{O}_L(x_i)) \tag{8}$$

$$\nabla_l = \mathbf{W}_{l+1}^T \nabla_{l+1} \odot \Psi'(\mathbf{O}_l(x_i)), \quad l = 1, 2, .., L-1 \tag{9}$$

where $\mathbf{Q}$ and $\mathbf{O}$ are intermediate functions used to simplify the equations, which can be formulated as:

$$\mathbf{Q}(\mathbf{O}_L(x_i)) \triangleq \frac{\mathbf{I}}{||\Psi_L(\mathbf{O}_L(x_i))||_2} - \frac{\Psi_L(\mathbf{O}_L(x_i)) \Psi_L(\mathbf{O}_L(x_i))^T}{||\Psi_L(\mathbf{O}_L(x_i))||_2^3} \tag{10}$$

$$\mathbf{O}_l(x_i) \triangleq \mathbf{W}_l \mathbf{h}_{l-1}(x_i) + \mathbf{B}_l, \quad l = 1, 2, \cdots, L \tag{11}$$

where $\mathbf{I}$ denotes an identity matrix. Updating rule for model parameters $(\mathbf{W}, \mathbf{B})$ using the SGD can be defined as:

$$\mathbf{W}_l := \mathbf{W}_l - \eta \frac{\partial \mathcal{L}_D}{\partial \mathbf{W}_l}, \quad l = 1, 2, .., L \tag{12}$$

$$\mathbf{B}_l := \mathbf{B}_l - \eta \frac{\partial \mathcal{L}_D}{\partial \mathbf{B}_l}, \quad l = 1, 2, .., L \tag{13}$$

where η is the learning rate. $Adam$ optimizer is used for adaptive learning rate decay with early stopping criterion selected empirically.

To effectively handle overfitting, we propose an adaptive learning rate algorithm. We define a learning rate decay mechanism as: $\eta_{t+1} = k \times \eta_t$, where η_t and η_{t+1} are the current epoch t and the next epoch $t+1$ learning rates, and k is the learning rate decay factor. We assign $\eta_0 = 0.001$ and $\eta_{min} = 0.0001$ as initial and minimum learning rates. After a certain number of epochs, validation loss may start increasing while training loss is still decreasing. This trend indicates model over-fitting. Therefore, we propose an early stopping criterion for the model training to avoid overfitting. Learning rate η_{t+1} is decayed with a factor of k if validation loss does not decrease for n epochs, where $n = 5$ is used in our experiments. If the learning rate approaches its lower bound, and validation loss does not further decrease, early stopping criteria will stop further training.

2.2. Proposed CNN based MTL Architecture

MTL in deep learning architectures is performed by introducing layers that are common to all the tasks. There are two important questions to be answered here i) when and how such parameter sharing could improve model and ii) what is the right balance between shared and task-specific parameters. In this work, we show that such inductive transfer can improve the model when the multiple tasks trained under a unified objective function with a specified task weighting scheme. We input the data to the multiple separate but identical branches. Each task-specific branch consists of three convolutional blocks followed by a flattening layer and final regression layer as shown in Fig. 1. By sharing initial layers of the model across the tasks, training examples force the parameters to be generalized in a manner such that model more compelling towards good weights yielding a better generalization of the learned parameters. Furthermore, the proposed architecture contains 2.4 times less number of trainable parameters as compared to PilotNet [11].

Table 4. Evaluation of estimation of the driving parameter (steering angle) in terms of RMSE on Udacity dataset [14].

MODEL	RMSE
PREDICT ZERO [13]	0.2076
3D LSTM [13]	0.1123
TRANSFER [13]	0.0709
PREDICT ZERO [15]	0.2077
PREDICT MEAN [15]	0.2098
ALEXNET [15]	0.1299
VGG-16 [15]	0.0948
ST-CONV + CONVLSTM +LSTM [15]	0.0609
PILOTNET	0.0986
AUTOPILOT	0.0831
PROPOSED	**0.0595**

3. RESULTS AND DISCUSSION

3.1. Weighting Scheme Analysis

We evaluate our proposed approach on three data sets, including Comma.ai [18], Udacity [14] and $GTA-V$ (see also Table. 1). Results on all three datasets are summarized in Table 2. It can be seen that the weighting scheme II (adaptive weighting) outperforms the other two schemes on all three datasets. In weighting scheme II the weighting factors were computed by taking the inverse of final validation loss (MSE) computed from previous training when a same unit weighting factor was assigned to each class. This introduced normalization among individual losses and eliminated the undesired bias due to a non-uniform range of values. A graphical comparison of all three weighting schemes is shown in Fig. 3. It can be seen that the proposed weighting scheme (II) comparatively converges more rapidly as well as this scheme has lower validation loss, which shows that the credibility of the proposed method.

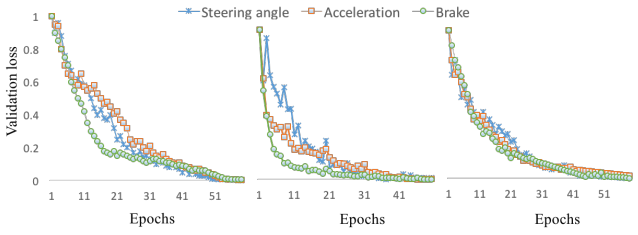

Fig. 2. Normalized validation losses of each driving parameter for the proposed multi-task model with configuration I (left), II (center), III (right) on Comma.ai dataset [18].

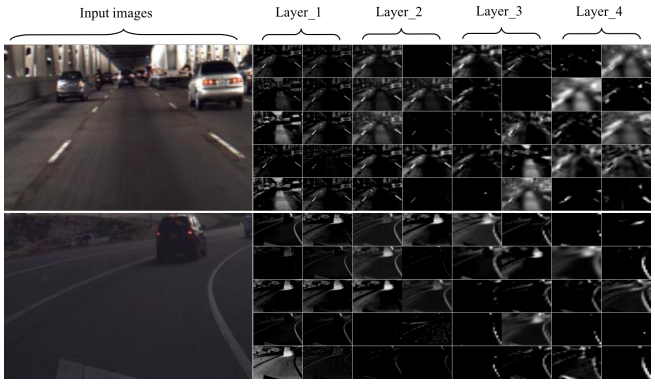

Fig. 3. Visualization of deep features extracted from activation layers of the proposed model.

3.2. Comparison with state-of-the-art models

Table 3 compares the performance of the proposed model and state-of-the-art models for a single driving parameter (steering angle) on Comma.ai dataset [18]. In [12], researchers proposed CNN+FCN model and evaluated their approach on Comma.ai dataset [18] for just a single task (steering angle) and achieved 2.44 and 3.20 MAE (mean absolute error) and standard deviation. In [12], researchers proposed a simple baseline end-to-end deep learning model and achieved 2.42 MAE with 3.26 standard deviation. In contrast to these approaches, our proposed method surpass these state-of-the-art models and achieved 1.106 MAE with 1.65 standard deviation. All compared approaches trained the models as single task problem whereas we solve this problem as multitask problem and our model predicts not only steering angle but two additional parameters.

In [13], the researcher used transfer learning, 3D LSTM, and baseline model (Predict zero) for a single task estimation. In 3D LSTM model, they tried to find saliency maps by feeding a video clip to the model and they find a change in saliency features in a sequence of frames and utilized these changes by collapsing multiple saliency maps as an image to map steering angle from these collapsed saliency maps. Using this approach, they achieved 0.1123 RMSE. In the transfer learning approach, they used ResNet50 pre-trained model and utilized saliency maps in the same manners as discussed earlier. Using the transfer learning model, they achieved 0.0709 RMSE on the test dataset. Another CNN+LSTM architectures proposed in [15] achieved 0.0609 RMSE. In the transfer learning approach, they achieved 0.948 RMSE on Udacity dataset. They compared their approach with two blind prediction schemes, Predict zero (always predict zero) and Predict mean (always predict mean). In contrast, our proposed model and task weighting scheme outperform all reported state-of-the-art models by achieving 0.0505 RMSE. In addition to that, our proposed model not only predict steering angle but also predict two additional driving parameters without affecting the performance of steering angle.

We also retrained PilotNet [11] and Autopilot [18] models for only single task (steering angle) on Udacity dataset [14] and achieved 0.0986 and 0.0831 RMSE for PilotNet [11] and Autopilot [18] models, respectively. In contrast, our MTL model outperforms all reported state-of-the-art models by achieving 0.0505 RMSE. In addition to that, our proposed model not only predict steering angle but also predict two additional driving parameters without affecting the performance of steering angle. Further, we observed that initial common layers learn common features of tasks such as road shape, lane markings and traffic as shown in Fig. 3. Visualization of activations on a query image supports the proposed soft sharing CNN architecture as sharing soft features among all the tasks can increase the performance of the model and also reduce the model complexity.

4. CONCLUSIONS

In this paper, we propose a task weighting scheme and soft parameter sharing for multi-task learning. Effect of task weighting scheme and the architecture of the proposed model encourages the learning of a dynamic range of multi-tasks. The proposed method is applied in the autonomous driving domain to simultaneously estimate steering angle, brake and acceleration using video frames captured by forward looking camera. Experiments are performed on two publicly available datasets: Comma.ai [18] and Udacity [14]. In addition to that, a new dataset has also been proposed using GTA-V gaming environment. In conclusion, we showed that training a single multi-task model with the proposed task weighting scheme can achieve competitive performance as compared to state-of-the-art methods. We believe our results provide useful insights in both the research and application of single-task and multi-task learning.

5. REFERENCES

[1] D. Coviello, A. Ichino, and N. Persico, "Time allocation and task juggling," *American Economic Review*, vol. 104, no. 2, pp. 609–23, 2014.

[2] I. Augenstein, S. Ruder, and A. Søgaard, "Multi-task learning of pairwise sequence classification tasks over disparate label spaces," *arXiv preprint arXiv:1802.09913*, 2018.

[3] I. Misra, A. Shrivastava, A. Gupta, and M. Hebert, "Cross-stitch networks for multi-task learning," in *International Conference on Computer Vision and Pattern Recognition*. IEEE, 2016, pp. 3994–4003.

[4] Z. Wu, C. Valentini-Botinhao, O. Watts, and S. King, "Deep neural networks employing multi-task learning and stacked bottleneck features for speech synthesis," in *International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2015, pp. 4460–4464.

[5] I. Kokkinos, "Ubernet: Training a universal convolutional neural network for low-, mid-, and high-level vision using diverse datasets and limited memory." in *International Conference on Computer Vision and Pattern Recognition*. IEEE, 2017, pp. 6129–6138.

[6] F. J. Bragman, R. Tanno, Z. Eaton-Rosen, W. Li, D. J. Hawkes, S. Ourselin, D. C. Alexander, J. R. McClelland, and M. J. Cardoso, "Quality control in radiotherapy-treatment planning using multi-task learning and uncertainty estimation," 2018.

[7] W. Xu, W. Liu, H. Chi, X. Huang, and J. Yang, "Multi-task classification with sequential instances and tasks," *Signal Processing: Image Communication*, vol. 64, pp. 59–67, 2018.

[8] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *International Conference on Computer Vision*. IEEE, 2017, pp. 2980–2988.

[9] A. Søgaard and Y. Goldberg, "Deep multi-task learning with low level tasks supervised at lower layers," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, vol. 2, 2016, pp. 231–235.

[10] A. K. C. Chen, A. Seff and J. Xiao, "Deepdriving: Learning affordance for direct perception in autonomous driving," in *International Conference on Computer Vision*. IEEE, 2015.

[11] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang *et al.*, "End to end learning for self-driving cars," *arXiv preprint arXiv:1604.07316*, 2016.

[12] J. Kim and J. F. Canny, "Interpretable learning for self-driving cars by visualizing causal attention." in *International Conference on Computer Vision*. IEEE, 2017, pp. 2961–2969.

[13] S. Du, H. Guo, and A. Simpson, "Self-driving car steering angle prediction based on image recognition," *Department of Computer Science, Stanford University, Tech. Rep. CS231-626*, 2017.

[14] Udacity, "https://www. udacity.com/self-driving-car , 2017," *publicly available*, 2016.

[15] L. Chi and Y. Mu, "Deep steering: Learning end-to-end driving model from spatial and temporal visual cues," *arXiv preprint arXiv:1708.03798*, 2017.

[16] M. Teichmann, M. Weber, J. M. Zöllner, R. Cipolla, and R. Urtasun, "Multinet: Real-time joint semantic reasoning for autonomous driving," *Intelligent Vehicles Symposium*, pp. 1013–1020, 2018.

[17] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *International Conference on Computer Vision and Pattern Recognition*. IEEE, 2016, pp. 770–778.

[18] E. Santana and G. Hotz, "Learning a driving simulator," *arXiv preprint arXiv:1608.01230*, 2016.

[19] Z. Chen and X. Huang, "End-to-end learning for lane keeping of self-driving cars," in *Intelligent Vehicles Symposium*. IEEE, 2017.