In this supplementary document, we first study the common failure cases of behavior cloning in dense urban scenarios. We then provide implementation details of the architecture and the training procedure used in the main paper. Further, we analyze the theoretical guarantees of the proposed modifications. Finally, we provide additional analysis of the results mentioned in the main paper. The supplementary video contains qualitative comparison of our approach DA-RB$^+$ again CILRS$^+$ in different weather conditions and traffic scenarios.

1. Failure Cases of Behavior Cloning

In this section, we study the common failure cases of behavior cloning in dense urban scenarios. For this, we consider the conditional imitation learning model [3] which is the current state-of-the-art on the CARLA 0.8.4 NoCrash benchmark. The common failure cases correspond to collision with pedestrians, collision with vehicles and traffic light violations. In most of these scenarios, we observe that the driving policy is not able to brake adequately as shown in the Fig. 4. We also provide driving videos of these scenarios in the attached video.

2. Implementation Details

In this section, we first give a detailed description of the architecture used in our approach. We then describe the loss function and the training protocol employed in our approach.

2.1. Architecture

We build on the conditional imitation learning framework of [3] and use the exact same architecture (Table 1) as that of CILRS [3] model. The input to the model consists of an image of resolution 200x88 and the current speed measurement. The image is processed by a ResNet34-based perception module resulting in a latent embedding of 512 dimension. The speed input is processed by two fully-connected layers of 128 units each and combined with the ResNet output using another fully-connected layer of 512 units. This joint embedding is then passed as input to the command branches and the speed branch which output the control values and the predicted speed, respectively. Each of these branches consists of two fully-connected layers with 256 units. We apply a dropout of 0.5 to the last fully-connected layer in each of the branches. CARLA [4] also provides access to four high level navigational commands - (i) turn left, (ii) turn right, (iii) go straight (at intersection) and (iv) follow lane. These high level commands are used as input to a conditional module which selects one of the four command branches to output the control, which consists of steer, throttle and brake.

2.2. Loss Function

The network is trained in a supervised manner with the loss function consisting of two components - (i) Imitation Loss: To imitate the expert actions, we use the $L_1$ loss between the predicted control $\pi(s)$ and the expert control $\pi^*(s)$. This

*indicates equal contribution, listed in alphabetical order
Module | Input | Output
--- | --- | ---
Measured Speed | 1 | 128
 | 128 | 128
 | 128 | 128
Joint Input | 512 + 128 | 512
Command branch | 512 | 256
 | 256 | 256
 | 256 | 3
Speed Prediction | 512 | 256
 | 256 | 256
 | 256 | 1

Table 1: Conditional Imitation Learning Architecture.

is represented as $L_{imitation} = \|\pi(s) - \pi^*(s)\|_1$. (ii) **Speed Loss**: Expert demonstrations have an inherent inertia bias, where most of the samples with low speed also have low acceleration. It is critical to not overly correlate these since the vehicle would prefer to never start after slowing down. This issue can be alleviated by predicting the current vehicle speed as an auxiliary task [3]. Therefore, we also use a speed prediction loss, given by $L_{speed} = \|v - \hat{v}\|_1$ where $v$ is the actual speed, $\hat{v}$ is the predicted speed and $\|\cdot\|_1$ denotes the $L_1$ norm. The final loss function is a weighted sum of the two components, with a scalar weight $\lambda$, given by $L = L_{imitation} + \lambda \cdot L_{speed}$.

2.3. Data Generation

We use the standard CARLA 0.8.4 data-collector framework\footnote{https://github.com/carla-simulator/data-collector} for generating data. We consider 4 weather conditions - 'ClearNoon', 'WetNoon', 'HardRainNoon' and 'ClearSunset' - for generating a total of 10 hours of expert training data and 2 hours of validation data in 'Town01' setting with the number of vehicles in the range [30, 60] and number of pedestrians in the range [50, 100]. The expert policy used in the data generation process consists of an A* planner followed by a PID controller and is provided by the official data collector. The images are rendered at a resolution of 800x600, and then processed at a resolution of 200x88 as in [3].

2.4. Training Protocol

We use the conditional imitation learning framework\footnote{https://github.com/felipecode/coiltraine} provided by the authors of [3] for training all methods mentioned in the paper. In all experiments, we use the Adam [6] optimizer and the exact same hyper-parameters as in the original CILRS [3] model. We save the model checkpoints after every 10000 iterations and stop training once the validation loss has stopped improving for 5 consecutive checkpoints. For all iterative algorithms mentioned in the paper, we initialize the behavior policy in each iteration with the trained policy of the previous iteration.

3. Theoretical Guarantees

DAgger [9] is known to have a better performance bound (Eq. (1)) on the total cost incurred over the time horizon compared to behavior cloning (Eq. (2)), which is given by

$$J(\pi) \leq J(\pi^*) + uT\epsilon_N + O(1), \quad \epsilon_N = \min_{\pi \in \Pi} \frac{1}{N} \sum_{i=1}^{N} E_{s \sim d_{\pi}} \ell(s, a)$$

$$J(\pi) \leq J(\pi^*) + T^2\epsilon, \quad \epsilon = E_{s \sim d_{\pi}} \ell(s, a)$$

where $J(\pi) = \sum_{t=0}^{T-1} \sum_{a \in A} [E_{a \sim \pi} (\ell(s, a))]$ is the total cost incurred by the policy $\pi$ over the time horizon $T$, $\ell(s, a)$ is a convex upper bound on the (in general non-convex) loss function $\hat{\ell}(s, a)$ and $u$ upper bounds $Q^\pi_t(s, a) - Q^\pi_t(s, \pi^*(s))$ for all $a \in A$, $s \in S$ and $t \in \{0, ..., T - 1\}$. 

---

\[\text{Table 1: Conditional Imitation Learning Architecture.}\]

\[\text{is represented as } L_{imitation} = \|\pi(s) - \pi^*(s)\|_1. \text{ (ii) **Speed Loss**: Expert demonstrations have an inherent inertia bias, where most of the samples with low speed also have low acceleration. It is critical to not overly correlate these since the vehicle would prefer to never start after slowing down. This issue can be alleviated by predicting the current vehicle speed as an auxiliary task [3]. Therefore, we also use a speed prediction loss, given by } L_{speed} = \|v - \hat{v}\|_1 \text{ where } v \text{ is the actual speed, } \hat{v} \text{ is the predicted speed and } \|\cdot\|_1 \text{ denotes the } L_1 \text{ norm. The final loss function is a weighted sum of the two components, with a scalar weight } \lambda, \text{ given by } L = L_{imitation} + \lambda \cdot L_{speed}.\]
Table 2: Performance Comparison of DA-RB$^+$ (E) and CILRS$^+$ for different Weathers. We report the number of successful episodes on all weathers in dense setting of all evaluation conditions. The evaluation consists of 25 episodes for each weather. However, as described in [1], $u$ in Eq. (1) may be $O(T)$, e.g., if there are critical states $s$ such that failing to take the action $\pi^*(s)$ in $s$ results in forfeiting all subsequent rewards. For example, in dense urban driving, these critical states correspond to scenarios involving close proximity to pedestrians and vehicles resulting in collision and termination of episode, so $u = O(T)$. In the presence of these type of scenarios, DAgger has a bound of $O(T^2)$ (Eq. (1)) which is the same as that of behavior cloning. Moreover, this bound can be improved to $O(T)$ (Eq. (3), see [1] for more details) by performing accurately on the critical states.

$$J(\pi) \leq J(\pi^*) + T\epsilon_N + O(1), \quad \epsilon_N = \min_{\pi \in \Pi} \frac{1}{N} \sum_{i=1}^{N} \mathbb{E}_{s \sim d_{\pi_i}} \ell(s, a)$$

The DA-CS variant of our approach explicitly samples the critical states during the aggregation process, thereby increasing the proportion of these states in the training data distribution leading to policies that perform better in these difficult scenarios. Therefore, assuming a convex upper bound on the loss function, DA-CS has better performance guarantees on the total cost incurred over the time horizon compared to DAgger.

While adaptive sampling methods in a mixture of distributions setting are known to have convergence guarantees [2], the theoretical analysis of the performance guarantees for the DA-RB variant of our approach is beyond the scope of this work.

4. Additional Experimental Results

4.1. Weather-wise Performance Breakdown

We provide the performance breakdown of DA-RB$^+$ (E) over individual weather conditions and compare against the CILRS$^+$ baseline. The evaluation consists of 25 episodes for each weather condition with different start locations and destinations. The results in Table 2 show that our approach outperforms CILRS$^+$ [3] in most of the conditions. However, MidRainSunset and HardRainSunset weather conditions are especially challenging for behavior cloning since none of the approaches are able to complete even a single episode out of 25. This is due to the presence of extreme conditions, such as excessive shadows, reflections of the buildings in water puddles and glare from sunset which severely complicates perception.

4.2. Considering Traffic Light Violation as Failure Case

In the NoCrash [3] benchmark traffic light violation is not considered as a termination scenario. However, traffic light violation can lead to fatal accidents especially in dense urban setting due to the presence of high number of pedestrians & vehicles. Therefore, obeying traffic lights is an essential part of urban driving which needs to be learned by the driving policy.
Environmental Conditions

Success Rate

0
10
20
30
40

Train NW NT NTW
CILRS+ DA-RB+

Figure 1: Success rate when considering traffic light violation as a failure case. NW - New Weather, NT - New Town, NTW - New Town & Weather.

In this experiment, we consider traffic light violation as a failure case and compare against CILRS+ model. We report the results in Fig. 1 on the dense setting on all the evaluation conditions. Our experiments show that our approach enables the policy to better detect traffic lights.

4.3. Comparison of SMILe against DAgger

We implement DAgger as per Algorithm 3.1 of [9]. In each iteration of DAgger, we append 2 hours of on-policy data to the current iteration dataset. For SMILe, we follow Algorithm 4.1 of [8] with $\alpha = 0.2$. We execute both algorithms using the same initialization for fair comparison. In our experiments (Fig. 3 in the main paper), we observe that the performance of SMILe is either as good as DAgger in New Weather and New Town & Weather conditions or slightly better in Training and New Town conditions. This is in contrast to the results in [9] where the authors show DAgger to be empirically superior to SMILe. We have shown in the main paper that DAgger is not effective for dense urban driving since the aggregation process does not address the dataset bias issue. However, the training dataset in each iteration in SMILe is sampled from a mixture of policies which leads to better diversity compared to direct aggregation. Also, we observe that SMILe+ generalizes very well to New Town and New Town & Weather conditions. This happens due to 2 reasons, (1) Triangular perturbations contribute to the diversity of the data since they simulate off-road drift which is seldom present in the expert’s state distribution, (2) SMILe returns an ensemble of policies trained in each iteration which leads to increased robustness and better generalization.

4.4. Comparison of DART against Triangular Perturbations

For implementing DART [7], we closely follow the code provided by the authors of [7]3. The performance of DART is quite similar to that of CILRS+ in most of the evaluation conditions (Fig. 3 and Table 3 in the main paper). DART uses a noise model which is optimized to iteratively minimize the covariate shift. These perturbations manifest most prominently in the steering of the vehicle, thereby simulating off-road drift. This is identical to the behavior modeled by triangular perturbations in the steering, therefore, leading to similar results.

4.5. Data Distribution Statistics for Different Sampling Methods

We report statistics regarding the data distribution induced by the sampling mechanisms (Section 3.3 and 4.4 in the main paper) to provide insights into the different type of scenarios captured by the sampling strategies. We focus on two type of statistics - (i) weather-wise data distribution over high level navigational commands (Fig. 2): We report the number of images in the training data for 'follow lane', 'turn left', 'turn right' and 'go straight (at intersection)' navigational commands, (ii) weather-wise data distribution over control values (Fig. 3): We bin the control values into 4 categories - 'brake', 'steer left', 'go straight' and 'steer right'. For the 'brake' category we consider the states where brake is emphasized by the expert policy. Even though brake can have a continuous value in the range $[0, 1]$, we observe that the brake distribution is highly skewed towards the extreme values. Moreover, while visualizing the driving performance, we notice that even a small value produces substantial braking effect. Therefore, we bin all the states where the brake $> 0.1$ in the 'brake' category. The other

3https://github.com/BerkeleyAutomation/DART
Figure 2: Weather-wise data distribution over high level navigational commands for the training data. We report the number of images in the training data for the sampling methods described in Section 3.3, 4.4 and Table 5 of the main paper - base 10 hours dataset (Base), absolute error on brake ($AE_b$), absolute error on steer, throttle and brake ($AE_{all}$), uncertainty-based sampling (Unc), ranking of expert states (Rank) and intersection & turning scenarios (IT).

3 categories are defined based on the steering values, which belong in the range $[-1, 1]$. We bin all the states where $\text{steer} < -0.1$ into 'steer left', $\text{steer} \in [-0.1, 0.1]$ into 'go straight' and $\text{steer} > 0.1$ into 'steer right' categories. We prioritize the 'brake' category over the steering categories since braking is the most crucial action to avoid collisions and other failure cases.

From the results in Table 5 of the main paper and Fig. 2, we can observe that the weather ‘ClearSunset’ and the navigational command ‘Go straight (at intersection)’ are most correlated with the generalization performance since the sampling method based on absolute error on brake ($AE_b$) results in the best generalization performance of the driving policy in terms of success rate. This is also apparent by the results of uncertainty-based sampling since it generalization performance is inferior compared to other sampling approaches. Furthermore, from Fig. 3, we can see that a uniform distribution over the control categories and training weathers results in the best generalization performance. Table 3 also provides additional insights into the inferior generalization performance of the uncertainty-based sampling approach. Even though uncertainty-based
Figure 3: Weather-wise data distribution over control categories for the training data. We report the number of images in the training data for the sampling methods described in Section 3.3, 4.4 and Table 5 of the main paper - base 10 hours dataset (Base), absolute error on brake ($AE_b$), absolute error on steer, throttle and brake ($AE_{all}$), uncertainty-based sampling (Unc), ranking of expert states (Rank) and intersection & turning scenarios (IT).

sampling is effective in capturing states where the brake is emphasized, its distribution is highly skewed. This results in the driving policy being overly cautious and braking excessively due to which the policy times out frequently and is not able to successfully complete the episode.

4.6. GradCAM Attention Maps

We examine the GradCAM [10] attention maps of DA-RB$^+$ qualitatively to visualize the region in the image which is important for vehicle control and compare against CILRS$^+$. Specifically, we backpropagate the gradients from the brake signal since it is very important for preventing collisions. The attention maps (Fig. 5) show that our approach enables the driving policy to focus more on the essential aspects of the scene, thereby learning a better implicit visual representation of the environment for urban driving.
Figure 4: **Common Failure Cases of Behavior Cloning in Urban Environments.** Top: collision with pedestrian. Middle: collision with vehicles. Bottom: traffic light violation. Note the major deviation in brake values compared to expert.
Figure 5: GradCAM Attention Maps.
References


[9] Stéphane Ross, Geoffrey J. Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In Conference on Artificial Intelligence and Statistics (AISTATS), 2011. 2, 4