Exploring Data Aggregation in Policy Learning for Vision-based Urban Autonomous Driving

Aditya Prakash\textsuperscript{1} Aseem Behl\textsuperscript{1,2} Eshed Ohn-Bar\textsuperscript{1,3} Kashyap Chitta\textsuperscript{1,2} Andreas Geiger\textsuperscript{1,2}

\textsuperscript{1}Max Planck Institute for Intelligent Systems, Tübingen \textsuperscript{2}University of Tübingen \textsuperscript{3}Boston University

\{firstname.lastname\}@tue.mpg.de

Abstract

Data aggregation techniques can significantly improve vision-based policy learning within a training environment, e.g., learning to drive in a specific simulation condition. However, as on-policy data is sequentially sampled and added in an iterative manner, the policy can specialize and overfit to the training conditions. For real-world applications, it is useful for the learned policy to generalize to novel scenarios that differ from the training conditions. To improve policy learning while maintaining robustness when training end-to-end driving policies, we perform an extensive analysis of data aggregation techniques in the CARLA environment. We demonstrate how the majority of them have poor generalization performance, and develop a novel approach with empirically better generalization performance compared to existing techniques. Our two key ideas are (1) to sample critical states from the collected on-policy data based on the utility they provide to the learned policy in terms of driving behavior, and (2) to incorporate a replay buffer which progressively focuses on the high uncertainty regions of the policy’s state distribution. We evaluate the proposed approach on the CARLA NoCrash benchmark, focusing on the most challenging driving scenarios with dense pedestrian and vehicle traffic. Our approach improves driving success rate by 16\% over state-of-the-art, achieving 87\% of the expert performance while also reducing the collision rate by an order of magnitude without the use of any additional modality, auxiliary tasks, architectural modifications or reward from the environment.

1. Introduction

Autonomous driving research has been gaining traction in industry and academia with the advancement in deep learning, availability of simulators [20, 24, 50] and large scale datasets [1,13,26,51,64,65]. While industrial research

*indicates equal contribution, listed in alphabetical order
state distribution encountered by the policy, learned policies quickly accumulate errors, leading to poor performance in new environments. This is referred to as the compounding error problem. DAgger [54] (Fig. 1) is a common data aggregation technique for learning policies that can better handle covariate shift and has been very effective in robotic tasks [5, 18, 42, 46, 55]. We perform an extensive analysis of DAgger for autonomous driving in CARLA [20] and find that the performance of DAgger starts to drop as the number of iterations increase, even in the training conditions. Moreover, we observe that simple hand-engineered modifications outperform DAgger in all the evaluation conditions. This indicates that the aggregated on-policy data contains redundant and non-informative states leading to sub-optimal performance. Therefore, we utilize a sampling mechanism to extract critical states from the generated on-policy data which pose high utility to the learned policy. While DAgger can guide the learning process of the driving policy, its aggregation process ignores potential issues in data-driven learning, specifically bias and overfitting to the aggregated data provided by the expert and the learned policy. As a result, we observe DAgger to fail when generalizing to new environments. To enable learning a more robust end-to-end policy, we propose to better guide the aggregation process in DAgger with a sampling mechanism and a replay buffer, and demonstrate significant gains.

Contributions: The primary contribution of our paper is a comprehensive analysis of data aggregation techniques for dense urban autonomous driving. We demonstrate the limitations of DAgger in terms of its inability to capture critical states and generalize to new environments and present a modified version of DAgger for collecting on-policy data for training driving policies. We propose to sample critical states from the on-policy data based on the utility they pose to the learned policy in terms of proper driving behavior and include a replay buffer which progressively focuses on the high-uncertainty regions of the learned policy’s state distribution. We experimentally validate that our approach enables the driving policy to achieve 87% of the expert performance and learn a better implicit visual representation of the environment for urban driving.

Our code and trained models are provided at https://github.com/autonomousvision/data_aggregation.

2. Related Work

Imitation Learning (IL): IL for self-driving has its roots in the pioneering work of [47]. IL uses expert demonstrations to directly learn a policy that maps states to actions [2, 3, 36, 49]. In contrast to modular [38], affordance-based [9, 57] reinforcement learning [33, 40], multi-task [39] and planning [8, 66] approaches, IL can be trained end-to-end in an off-line manner with expert data collected in the real world or a simulated environment. More recently, Codevilla et al. [11, 12] proposed a conditional IL framework by utilizing high-level directional commands and show that these models perform well in urban scenarios.

IL for sequential decision making tasks is addressed as a supervised learning problem in which the policy is trained under the state distribution induced by expert. However, this is non-optimal since the learned policy influences the future states that it encounters, which can be different compared to the expert’s state distribution. This phenomenon, referred to as covariate shift [54], leads to the compounding error problem. In the context of dense urban driving, this is even more prominent due to non-deterministic behavior of dynamic agents. This problem can be addressed using iterative on-policy [4, 5, 30, 52–54, 60] and off-policy [34] methods, which we discuss next. We build upon these in the conditional imitation learning framework and propose modifications that lead to better empirical results.

DAgger: DAgger [54] is an iterative training algorithm that collects on-policy data at each iteration based on the current policy and trains the next policy on the aggregate of collected datasets. Several variants of DAgger have been proposed such as Q-DAgger [4], AggreVaTe [53], AggreVaTeD [60], DAggerFM [5], SafeDAgger [67], MinDAgger [44], which focus on improving sample complexity [5, 44, 60, 67] and minimizing cost-to-go of the expert [53] or the policy [4]. DAgger has also been explored in the context of autonomous driving [10] in off-road driving scenario [46] and TORCS racing simulator [67]. However, we show that a direct application of DAgger is not optimal for dense urban driving and propose modifications that lead to better empirical results. In this regard, Q-DAgger [4] and minDAgger [44] are most related to our work since they also highlight the limitations of the training data distribution induced by DAgger. While the former focuses on decision tree policies for verifiability and the latter focuses on data efficiency for discrete policies in static Minecraft environments, we investigate DAgger and its variants for end-to-end continuous driving policies in highly dynamic urban environments.

SMILE: The Stochastic Mixing Iterative Learning Algorithm (SMILE) [52] allows the learner to retrain under the new state distribution induced by mixture of policies as it is updated across successive iterations. It defines an efficient dataset construction algorithm for the new state distribution at each iteration using a sampling mechanism over a mixture of policies, where the sampling proportion is independent of the policies. In contrast, our approach can be considered as an adaptive version of SMILE where the sampling proportion is dependent on the policies.

RAIL: Reduction-based Active Imitation Learning [30] (RAIL) is an iterative training method that uses active learning algorithms to sample from on-policy data to improve
sample complexity of the training dataset. Our approach is similar, in principle, to RAIL but our focus is on improving performance rather than sample complexity. We explore different sampling mechanisms and show that a variant of RAIL fails on our task. Furthermore, we present a simpler alternative which works better in practice.

**DART: DART** [34] is an iterative off-policy data perturbation approach which optimizes a noise model to minimize covariate shift. However, we show that DART is not effective in the case of autonomous driving since it is computationally expensive and similar performances can be achieved using hand-engineered perturbations. Instead, we focus on iterative on-policy learning which leads to better empirical results.

**Critical States:** A major challenge in sequential decision making tasks is to facilitate effective exploration of critical states [28] which are central for the policy to learn appropriate task specific behavior. Several notions of critical states based on mutual information [27,43], uncertainty [25,37,58], reducing expected error [56,62], diversity [14–17,29] and maximizing expected label changes [25,31,61] have been effectively applied in computer vision [32,41,48,59,61]. In the context of dense urban driving, the critical states constitute scenarios like proximity to vehicles and pedestrians, following traffic regulations etc. These are crucial since even a single failure can lead to fatal accidents. Therefore, an effective exploration strategy for these critical states is required to enable the driving policy to learn safe driving behavior. We explore different sampling mechanisms to incorporate these critical states into our approach.

### 3. Method

In this section, we first describe imitation learning in the context of autonomous driving. We then describe the original Dataset Aggregation (DAgger) algorithm, followed by our modifications that lead to significant performance gains.

#### 3.1. Imitation Learning for Autonomous Driving

The goal of imitation learning (IL) is to learn a policy \( \pi \) that imitates the behavior of an expert policy \( \pi^* \):

\[
\text{IL} : \arg\min_{\pi} \mathbb{E}_{s \sim P(s|\pi)} [\mathcal{L}(\pi^*(s), \pi(s))]
\]  

(1)

where \( P(s|\pi) \) represents the state distribution induced by driving policy \( \pi \) and \( \mathcal{L}(\cdot) \) represents the loss function. In our autonomous driving application, the output of the policy is a 3-dimensional continuous action vector (steer, throttle and brake of the car) and we use an \( L_1 \) loss for training.

The most simple approach for IL is Behavior Cloning (BC) which is a supervised learning approach. In this method, an expert policy is first rolled out in the environment to collect observations \( s^* \) of all visited states and the expert actions \( a^* \). The policy \( \pi \) is trained in a supervised manner using the collected dataset of state-action pairs:

\[
\text{BC} : \arg\min_{\pi} \mathbb{E}_{(s^*, a^* \sim P^*)} [\mathcal{L}(a^*, \pi(s^*))]
\]  

(2)

where \( P^* \) represents the state distribution provided by expert policy \( \pi^* \) and \( \mathcal{L} \) represents the loss function.

Behavior cloning assumes the state distribution to be i.i.d. since the next state is sampled from the states observed during expert demonstration which is independent from the action predicted by the current policy. This leads to the compounding error problem where the policy is unable to recover from its mistakes when it encounters a state that is not present in the expert’s state distribution. This problem can be solved using iterative on-policy algorithms such as DAgger which we discuss next.

#### 3.2. Dataset Aggregation (DAgger)

DAgger is an iterative training algorithm that collects on-policy trajectories at each iteration under the current policy and trains the next policy under the aggregate of all collected trajectories. The policy used to sample trajectories at each iteration can be represented as \( \tilde{\pi} = \beta \pi^* + (1 - \beta) \hat{\pi} \) where \( \pi^* \) is the expert policy and \( \hat{\pi} \) is the learned policy. Typically, \( \beta_0 = 1 \) and is decreased in successive iterations. DAgger effectively appends the current dataset with a set of input states that the learned policy is likely to encounter during its execution based on previous experiences. This mitigates the compounding error problem in progressive iterations since the agent now has supervision from the expert for the states where it deviates from the optimal behavior.

#### 3.3. Critical States

The DAgger algorithm appends the entire generated on-policy trajectory to the training dataset for the current iteration. However, not all states in the trajectory present the same utility for the driving policy. Specifically, states that correspond to failure cases of the driving policy are most relevant since they have maximum utility from the perspective of learning safe driving behavior. Therefore, we explore different mechanisms for sampling these critical states.

**Task-based:** In the context of dense urban driving, tasks such as making turns on intersections are more important than driving straight on an empty road since most of the collisions occur at intersection and turnings. CARLA provides access to high level navigational commands - (1) turn left, (2) turn right, (3) go straight (at intersection) and (4) follow lane. For task-based sampling, we ignore the on-policy data collected for ‘follow lane’, focusing on the other three situations, hence prioritizing sampling of intersections and turns. We assign equal importance to (1), (2) and (3).

**Policy-based:** For policy-based sampling, we use the epistemic uncertainty in the prediction of the driving policy to
sample critical states. To measure epistemic uncertainty, we use test-time dropout with probability 0.5 and calculate the variance in the predicted control [28]. The set of critical states \( S_c \) is then given by

\[
S_c = \left\{ s_c \in S \mid H(s_c, \pi, \pi^*) > \alpha \cdot \max_s H(s, \pi, \pi^*) \right\}
\]

where \( S = \{ s \mid s \sim P(s|\pi) \} \) is the set of states sampled from the state distribution \( P(s|\pi) \) and \( H(s, \pi, \pi^*) = \text{Var}(\pi(s)) \) denotes the sampling criterion with \( \text{Var}(\cdot) \) the dropout variance over \( \pi \) and \( \alpha < 1 \) chosen empirically.

**Policy and Expert-based:** In the presence of on-policy expert supervision, we explore multiple strategies: (a) We sample the on-policy states with the highest loss \( L(\cdot) \), thereby enforcing that the policy learns from its mistakes. More formally, we obtain the set of critical states \( S_c \) in Eq. (3) using \( S = \{ s \mid s \sim P(s|\pi) \} \) and \( H(s, \pi, \pi^*) = L(\pi, \pi^*) \). (b) We rank the expert states based on the loss incurred by the driving policy and sample the required proportion of states with the highest loss. Here, we set \( S = \{ s \mid s \sim P(s|\pi^*) \} \) and \( H(s, \pi, \pi^*) = L(\pi, \pi^*) \) in Eq. (3). (c) We observe that most of the failure cases like collisions and traffic light violations occur due to the inability of the driving policy to brake adequately. Thus, we sample based on deviations in the brake signal to identify these failure cases. For this, we use \( S = \{ s \mid s \sim P(s|\pi^*) \} \) and \( H(s, \pi, \pi^*) = L_b(\pi, \pi^*) \) in Eq. (3) where \( L_b \) denotes the (one-dimensional) brake component of the loss \( L \).

### 3.3. Implementation Details

We build on the conditional imitation learning framework\(^2\) of [12] using the open source CARLA simulator. We make no changes to the architecture (ResNet 34-based model) and use the code base provided by the authors of [12]. We initialize the policy with a behavior cloning policy trained on 10 hours of expert data. The size of the replay buffer is kept fixed at 10 hours. At each iteration, we generate \( \sim 15 \) hours of on-policy trajectories and sample critical states using the previously defined methods. We set the threshold \( \alpha \) for sampling such that we generate \( \sim 2 \) hours in the first iteration and keep it fixed in subsequent iterations. Consequently, as the policy gets better in each iteration, the total proportion of sampled on-policy data decreases since the threshold is fixed. We terminate the algorithm when the total proportion of sampled trajectories from the generated on-policy data falls below a predefined threshold, set as 0.5 hours. At this stage, we can say that the policy has learned proper driving behavior since the failure cases constitute very low proportion of the generated on-policy trajectories and we use this policy for evaluation. More details are provided in the supplementary and code.

### 4. Experiments

We conduct three types of experiments to validate our approach. First, we analyze the driving performance of the learned policy in dense urban setting and compare against

---

\(^{1}\) Refer to the supplementary for theoretical analysis

\(^{2}\) https://github.com/felipecode/coiltraine
several baselines. Second, we conduct an infraction analysis to study different failure cases. Finally, we present a variance analysis to examine the robustness of our proposed approach against random training seeds.

**Baselines:** For analyzing the driving performance, we compare our method against CILRS [12], DAgger [54], SMILE [52] and DART [34] baselines. CILRS is the current state-of-the-art on the NoCrash benchmark on CARLA 0.8.4. We run all algorithms under 2 initializations - policy trained with 10 hours of expert no-noise data and policy trained with 10 hours of expert data with 20% triangular perturbations [12] (denoted by *). All the algorithms used in our experiments are shown in Table 1. We follow Algorithm 3.1 of [54] and algorithm 4.1 of [52] for implementing DAgger and SMILE respectively. For DART, we closely follow the code provided by the authors of [34]. For our infraction analysis, we focus on CILRS [12] since it is significantly better compared to other approaches and serves as a strong baseline. For our variance study, we compare our approach against CILRS [12] and DAgger [54].

**Dataset:** We use the CARLA [20] simulator as the environment for training and evaluation, specifically CARLA 0.8.4 which consists of two towns - Town 1 and Town 2. We consider the dense urban setting of the challenging NoCrash benchmark as our evaluation setting since it accurately represents the complexities of urban driving. The driving policy is trained with data collected in Town 1 with 4 different weather and evaluated across different environments - Training, New Weather (NW), New Town (NT) and New Town & Weather (NTW). The NoCrash benchmark consists of 2 new weather conditions. Instead, we report results on all 10 new weather conditions for a comprehensive evaluation of generalization ability. Therefore, our results cover a total of 4 training conditions and 24 generalization conditions of varying difficulty.

**Metrics:** For evaluation, we use the number of successfully completed episodes out of 100 (success rate) and infraction related metrics. We consider 4 possible cases of failure - collision with pedestrians, collision with vehicles, collision with other static objects and timed out scenarios. For our variance study, we report the standard deviation on the success rate based on 5 random training seeds.

### 4.1. Driving Performance

**DAgger:** In this experiment, we try to examine if on-policy data helps to improve driving performance, and see how it fares when compared against triangular perturbations. From Fig. 2, we observe that DAgger leads to improvement when compared to no-noise model but achieves similar performance as triangular perturbations. Moreover, the performance of DAgger starts to drop after the second iteration in the training conditions. This happens because as DAgger continues to append on-policy data, the diversity of the dataset does not grow fast enough compared to the growth of the main mode of demonstrations, e.g., driving straight in lane. Consequently, the performance decreases as more data is collected since the driving policy is not able to learn how to react in rare modes, e.g., close proximity to dynamic agents. This result is in direct contrast to prior applications of DAgger in robotics [5, 18, 42, 36, 55] and reflects the limitation of DAgger in case of datasets having significant bias. This observation is also consistent with [12] where the authors show that additional data does not necessarily lead to improvement in performance for urban autonomous driving. Further, we observe that the performance of DAgger in the generalization conditions starts to drop after the second iteration. This is expected since the aggregated on-policy data is collected in the training conditions, thereby leading to overfitting as the dataset size increases.

**DAgger with Critical States (DA-CS):** In this experiment, we evaluate our first modification to examine if it is able to mitigate the aforementioned issues. For the purpose of subsequent analysis, we use deviation in brake as the sampling mechanism since we observe that in most of the failure cases, the policy is not able to brake adequately. The results are shown in Table 2. In contrast to DAgger, DA-CS significantly outperforms triangular perturbations in training conditions, thereby affirming that the sampled critical states contain useful information that facilitate improved driving behavior. However, on the new weather condition, the performance of DA-CS starts to decline. This indicates that the policy is starting to overfit to the training conditions. Next, we evaluate our second modification to alleviate this issue.

**DAgger with Replay Buffer (DA-RB):** The goal of this experiment is to examine if the proposed replay buffer is able to alleviate the aforementioned overfitting problem. The results reported in Table 2 clearly show that the replay buffer helps to improve performance on new weather thereby helping generalization. This reflects the importance of controlling the proportion of expert data and on-policy critical states while training the driving policy. We further try to examine if the improved behavior due to triangular perturbations is complementary to improved behavior due to DA-RB. This is reflected in the increase in the success rate of DA-RB* compared to DA-RB (Table 2). This happens because the triangular perturbations model the drift of the policy along the lateral direction, e.g., moving off road whereas DA-RB focuses on the failure cases of the policy in the longitudinal direction, e.g., collision with pedestrians and vehicles, traffic light violations. By incorporating both kinds of behavior in the training dataset and utilizing expert supervision on these states, our approach enables the policy to learn accurate driving behavior, thereby alleviating
<table>
<thead>
<tr>
<th>Model</th>
<th>Iterative</th>
<th>Off-Policy</th>
<th>On-Policy</th>
<th>Perturbations</th>
<th>Aggregation</th>
<th>Sampling</th>
<th>CS</th>
<th>RB</th>
<th>Ensemble</th>
</tr>
</thead>
<tbody>
<tr>
<td>CILRS</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CILRS⁺</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DART</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DAgger</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DAgger⁺</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SMILe</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SMILe⁺</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DA-CS</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DA-RB</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DA-RB⁺</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DA-RB⁺(E)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 1: Different algorithms used in our experiments. CS - Critical states, RB - Replay Buffer, Gray - our methods.

Figure 2: Success rate of different methods across conditions. ‘⁺’ represents training with perturbed expert data.

the compounding error problem to a significant extent. We provide driving videos of these scenarios in supplementary.

Comparison against CILRS, DAgger and SMILe on all conditions: While all approaches are able to perform similar to CILRS⁺ on training conditions, we observe that most of them fail to generalize to new environments as evident by the drop in performance in Fig. 2. In contrast, DA-RB⁺ shows significant improvement against other methods when generalizing to NW and NT conditions. While it does not improve the success rate in NTW condition, it shows better overall driving behavior, as reflected in the collision metrics (Fig. 3). Further, we also evaluate an ensemble model of all DA-RB⁺ iterations (DA-RB⁺(E)). The results (Table 3) clearly show that ensemble helps in better generalization.
Comparison against DART: In this experiment, we examine if iterative off-policy perturbations outperform iterative on-policy approaches. In Fig. 2, we observe that DART achieves similar performance to DAgger and SMILe on most conditions, which is consistent with the results of [34]. However, DA-RB outperforms it significantly which shows that on-policy algorithms are more adept at handling covariate shift. This happens because critical states such as close proximity to dynamic agents are not present in the expert’s state distribution due to which off-policy approaches are not able to learn appropriate response to these scenarios.

Comparison against Expert: Since our approach does not make use of any additional modality, auxiliary task or reward from the environment, the performance of the trained policy is upper bounded by that of the expert. In this experiment, we examine if our approach facilitates maximum exploitation of the information contained in the data under the given constraints. The results in Table 3 show that DA-RB+(E) is able to achieve $\sim 87\%$ of the expert’s performance over all evaluation conditions. This shows that our approach enables the policy to learn accurate driving behavior. The expert results in Table 3 also highlight the challenging nature of driving in CARLA’s dense setting. This is due to non-deterministic and non-optimal behavior of dynamic agents which leads to increased collisions and timed out scenarios where multiple vehicles clog the road resulting in very little room for driving.

4.2. Infraction Analysis

The goal of this experiment is to evaluate the qualitative driving behavior of the learned policy which is reflected accurately in terms of infractions. We consider 4 types of infractions - collision with pedestrians, vehicles, other static objects and timed out cases. We report the number of failed episodes due to these infractions in NTW condition since this helps to evaluate the qualitative behavior with respect to generalization to new environment.

The results are shown in Fig. 3. We observe that DA-RB+ leads to significant reduction in collision with dynamic agents compared to CILRS+. This indicates that qualitative driving behavior of our model is superior to CILRS+. We also report the number of episodes which failed due to time out. While the major failure case in case of CILRS+ is collision with vehicles, the policy trained with our approach mostly gets timed out. This happens due to 2 reasons: (1) since our agent is better at obeying traffic lights, it stops for 5-8 seconds on an average in case of a red light which significantly increases the probability of getting timed out, (2) multiple vehicles clog the lane resulting in very little room for driving. In contrast, CILRS+ frequently collides with dynamic agents and violates traffic lights leading to reduced timed out cases but significantly higher collisions. This shows that our approach enables the policy to focus on the essential aspects of the scene, thereby learning a better implicit representation of the urban environment.

4.3. Training Seed Variance

We further examine the robustness of the learned policies wrt. variance in the training seed, a common problem in imitation learning [12]. For fair comparison, we use the same 10 hours of expert data as base data for all approaches and initialize the perception module with the weights of a network pre-trained on ImageNet [12] in all cases. This re-

<table>
<thead>
<tr>
<th>Task</th>
<th>CILRS+</th>
<th>DART</th>
<th>DA-RB+</th>
<th>DA-RB+(E)</th>
<th>Expert</th>
</tr>
</thead>
<tbody>
<tr>
<td>Train</td>
<td>$45 \pm 6$</td>
<td>$50 \pm 1$</td>
<td>$62 \pm 1$</td>
<td>$66 \pm 5$</td>
<td>$71 \pm 4$</td>
</tr>
<tr>
<td>NW</td>
<td>$39 \pm 4$</td>
<td>$37 \pm 2$</td>
<td>$60 \pm 1$</td>
<td>$56 \pm 1$</td>
<td>$72 \pm 3$</td>
</tr>
<tr>
<td>NT</td>
<td>$23 \pm 1$</td>
<td>$26 \pm 1$</td>
<td>$34 \pm 2$</td>
<td>$36 \pm 3$</td>
<td>$41 \pm 2$</td>
</tr>
<tr>
<td>NTW</td>
<td>$26 \pm 2$</td>
<td>$21 \pm 1$</td>
<td>$25 \pm 1$</td>
<td>$35 \pm 2$</td>
<td>$43 \pm 2$</td>
</tr>
</tbody>
</table>

Table 3: Success rate on dense setting of all conditions. Mean and standard deviation over 3 evaluation runs. NW=New Weather, NT-New Town, NTW-New Town & Weather, DA-RB+(E) - ensemble of DA-RB+ over all iterations.

Failure case analysis. We consider collision with pedestrians, vehicles, other static objects and timed out scenarios on the dense setting of New Town & Weather.

Figure 3: Failure case analysis. We consider collision with pedestrians, vehicles, other static objects and timed out scenarios on the dense setting of New Town & Weather.
duces the variance due to data collector and random initialization of the policy parameters, thereby ensuring that the primary source of variance is randomness in the training seed, in addition to the evaluation variance which is caused by the random dynamics in the simulator. We train the behavior cloning policy with 5 random training seeds for each of the approaches and report the standard deviation on success rate on the dense setting of New Town & Weather.

The results in Table 4 show that DA-RB$^+$ reduces the standard deviation due to random training seeds in successive iterations. This indicates that sampling the dataset based on critical states is crucial for variance reduction. In each iteration, we selectively sample critical states from a mixture of distributions induced by the trained policies in each of the previous iterations. In this context, Borsos et al. [7] have previously shown that mixture of distributions with adaptive importance sampling is effective in reducing variance of online learning algorithms and our results validate this theory in the context of urban autonomous driving.

### 4.4. Different Methods for Sampling Critical States

In this experiment, we present a comparative analysis of different sampling methods$^3$ (Section 3.3) to identify critical states. We consider 5 sampling methods - (1) Absolute Error on brake, $AE_b$ (2) Absolute Error on all control parameters (steer, control, brake), $AE_{all}$ (3) Uncertainty in policy’s predictions, Unc, (4) Ranking of expert states while sampling, Rank and (5) Intersection and turning scenarios, IT. To determine uncertainty, we run 100 instances of model with dropout $= 0.5$ and compute the variance in the predicted control. We initialize all methods with a policy trained on 10 hours of perturbed expert data (Base).

From Table 5, we observe that $AE_b$ performs best on most of the conditions indicating that brake is able to capture critical states required for urban driving. This happens because deviation in brake is able to capture instances where the agent is running a red light or approaching a pedestrian or vehicle at very close distance, which are most informative for urban driving. $AE_{all}$ is not as effective as brake since it averages out the deviation in the controls. For example, a deviation of $\delta$ in each of the three controls and a deviation of $3\delta$ in just the brake will both result in a mean of $\delta$ but the latter is more likely to lead to failure cases and hence more important. Our implementation of uncertainty-base sampling (Unc) corresponds to a variant of RAIL [30] with Query-Based Committee (QBC) as the active learning algorithm where the committee consists of 100 instances of behavior cloning policy with test-time dropout. This approach does not take into account any task-based or infraction-based information which leads to sub-optimal performance. This indicates that high uncertainty in prediction does not correlate with critical scenarios. Furthermore, selectively sampling expert states (Rank) does not lead to any improvement over on-policy data sampling, indicating that the latter contains critical states relevant for improved urban driving. Moreover, most of the collisions and traffic light infractions occur at the intersections, therefore, sampling the intersection & turning scenarios leads to significant improvement compared to the Base model.

### 5. Conclusion

In this paper, we conduct a rigorous study of on-policy data aggregation and sampling techniques in the context of dense urban driving in CARLA. We empirically show that DAgger is not optimal for this task and does not generalize well to new environments. We propose two modifications to the DAgger algorithm to alleviate the aforementioned issues. Experiments demonstrate that our approach enables the policy to generalize to new environments, reduces variance due to training seeds and helps in learning a better implicit visual representation of the environment for dense urban driving. Based on our findings, we expect an extensive study of active learning algorithms for autonomous driving to be a promising direction for future research.

### Acknowledgements

This work was supported by the BMBF through the Tübingen AI Center (FKZ: 01IS18039B). The authors also thank the International Max Planck Research School for Intelligent Systems (IMPRS-IS) for supporting Kashyap Chitta and the Humboldt Foundation for supporting Eshed Ohn-Bar.

---

$^3$Refer to the supplementary for statistics regarding data distribution

<table>
<thead>
<tr>
<th>Task</th>
<th>Base</th>
<th>$AE_b$</th>
<th>$AE_{all}$</th>
<th>Unc</th>
<th>Rank</th>
<th>IT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Train</td>
<td>36</td>
<td>50</td>
<td>50</td>
<td>39</td>
<td>51</td>
<td>55</td>
</tr>
<tr>
<td>NW</td>
<td>40</td>
<td>57</td>
<td>48</td>
<td>36</td>
<td>54</td>
<td>51</td>
</tr>
<tr>
<td>NT</td>
<td>18</td>
<td>33</td>
<td>30</td>
<td>23</td>
<td>23</td>
<td>33</td>
</tr>
<tr>
<td>NTW</td>
<td>24</td>
<td>26</td>
<td>28</td>
<td>27</td>
<td>26</td>
<td>23</td>
</tr>
</tbody>
</table>

---

Table 4: **Training Seed Variance.** Standard deviation of the success rate wrt. 5 random training seeds on the dense setting of New Town & Weather. Note that CILRS$^+$ is a non-iterative approach.

Table 5: **Success rate of different sampling methods on dense setting of all conditions.** Unc - Uncertainty based sampling, IT - Intersection & Turnings, NW - New Weather, NT - New Town, NTW - New Town & New Weather.
References

[6] Mariusz BojarSKI, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D. Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, Xin Zhang, Jake Zhao, and Karol Zieba. End to end learning for self-driving cars. arXiv.org, 1604.07316, 2016. 1
[28] Sandy H. Huang, Kush Bhatia, Pieter Abbeel, and Anca D. Dragan. Establishing appropriate trust via critical states. In


