Assignment 1

Farrin Marouf Sofian

April 2023

Github Repo

1 Part 1: Pen and Paper

1. **In imitation learning or reinforcement learning, what is the goal of policy network?**

   The goal of the policy network in both imitation learning and RL is to map states to actions. In imitation learning, the policy is trained to mimic expert decisions. The objective for the agent in RL is to learn to take actions that maximize the total reward, therefore the policy defines the agent's behaviour.

2. **What does it mean for a state to be partially or fully observable. Give some examples for each.**

   A state is fully observable if the agent has access to all information about its surrounding or can detect them in order to take the optimal action a^*. An example of this is board games or chess, where the agent has access to the rules and the position of the items.

   A partially observable state is one in which the agent does not have access to all the information about the environment or the environment is noisy or information is missing. An example of this is autonomous cars where they can only observe a portion of the environment and make decisions in noisy environments.

3. **What is the main assumption in Behavior Cloning? Explain how this assumption leads to the distribution shift problem**

   In Behaviour cloning, state distribution P^* is given to the expert. The main assumption is that the states are IID (Independent and Identically Distributed) meaning that the next state is sampled independently from the action performed by the current policy. This reduces the ability of the expert in generalizing to states that it has not observed before (novel states), in particular when the distribution is shifted it cannot perform well.

4. **Describe the two solutions that we have seen in the class for the distribution shift problem**

   We saw PilotNet which they used data augmentation. Using, additional cameras that were able to capture different angles, from left and right which gave them more information about the environment. Moreover, the viewpoint for e.g left camera would be used as an additional information for the e.g front camera while training. We saw DAgger (Dataset aggregation) their main idea was to make the distributions p_{π_θ} closer to p_{data} by collecting data from p_{π_θ} and running the policy to generate additional states, which are then labeled by an expert for their corresponding actions. Finally, the new data is aggregated with the original dataset.

5. **What is the problem of behavior cloning at the intersections? What is the solution that we have seen in the class to this problem? How does conditional imitation learning improve behavior cloning in these cases?**

   At intersections, there's an uncertainty since the probability of turning right, left or going straight is equal. To overcome this situation, CIL provides additional condition (direction) to the policy to make decision on. It improves the agent by limiting the space of actions (right, left, straight .etc) at the intersections. Also, they used noise injection which was a type of data augmentation and gave them a performance boost and help with behavior cloning.

6. **What is the role of affordances in self-driving? What kind of affordances can be useful for highways driving or inner-city driving or both? Compare the two by suggesting some specific example affordances for each case.**

   its the set of low-dimensional actions that is provided based on the observations from the environment that limits the space of actions. For highway driving since it is solely driving and no complex situations like pedesterians or intersections, the affordances can be speed, angle and distance to the lane marking etc. These affordances would both limit space of actions and also ensure that the car is within the lane marks and support swithing lanes. For inner-city driving the situation is different. There, we have intersections and pedesterians at the scene, hence speed, the state of traffic light, hazard stop can be useful.

7. **What are ideal properties of visual abstractions in self-driving?**
   1. Invariant: exclude unnecessary and irrelevant to details that we don't care (the policy).

   2. Universal: Policy should work for different environments (different weathers, urban, highway, etc.)

   3. Data efficient: if represetnation takes long to extract, it is not affordable, it should be efficient to use. (memory/computation efficient)

4. Label efficient: should contain labels, so that we don't spend a lot of time labeling.

8. **When we use semantic segmentation as an intermediate representation, we learned that fewer classes and coarsely annotated images perform better in self-driving. What could be the reason for that?** With fewer classes, the network has to learn to distinguish between fewer objects or regions, which can make the task easier and faster to learn. Coarser annotations can also simplify the task, as the network has to make fewer distinctions between similar-looking objects or regions and hence can generalize better.

9. **In visual abstractions, we said that typically $n_r << n_a$ where n_r is the number of images annotated with semantic labels and n_a is the number of images annotated with expert actions. Besides the cost of labelling, is there any other intuition behind this assumption?**

   Visual abstractions aim to capture the fundamental structure of a scene in a concise and easy-to-understand manner. Although semantic labels represent the most important characteristics of a scene, they may not provide significant information as they tend to be consistent across multiple images. Conversely, expert actions are unique to each image and demand a large number of annotations to cover the full range of actions that could take place. Having a collection of different action sets for each instance can lead to better generalization.

10. **What is the difference between online and offline evaluation? Why does relying on offline evaluation only can be dangerous? Why do offline metrics fail to represent online metrics?**

    In online evaluation, the agent gets to interact with the environment, observe the environment and make a prediction and then apply on the environment. However, this is expensive and not very safe to test out in the real world. On the other hand, offline evaluation is when the model makes a prediction based on the observation (from the dataset) and then an error is calculated between the models prediction and pre-defined ground truth. This is usual computer vision task. Since, online evaluation is based on interaction with the environment directly, it covers cases that may not be seen or covered in the dataset (offline mode0 hence, there is a difference in these distributions and hence solely training the model on offline data can be dangerous as it does not cover all possible scenarios.

2 Imitation Learning

In this section I implemented the CILRS paper and trained it for the given dataset, below I will give info on the architecture, training and results.

2.1 Architecture

Similar to the paper, I used a Resnet18 backbone (removed the last layer to extract features only) along with MLP's for measure speed, speed prediction and conditional modules. There are 4 conditional modules, each targetting a command which ranges from 0 to 3, indicating left, right, straight and lane follow. I applied dropout of 50% for MLP's and Relu activation after hidden layers. As stated in the paper, the image features are 1) fed into the speed prediction MLP to predict the target speed 2) it is concatenated with the output of the measured speed and fed into the conditional modules depending on the target command. The result of the final step is a set of actions (throttle, steer, brake).

2.2 Training

I trained the model for 10 and 30 epochs with batch size of 128 and learning rate of 2e-4, Adam optimizer with weight decay of 1e-4. As for the loss function , I tested with L1 and L2 loss but as stated in the paper as well, L2 loss performed better. As suggested in the CIL paper I downscaled the speed (multiplied with 0.5). The results varied depending on how many epochs and which losses I used. I will provide detailed loss function figures for each and provide details. without dropout slightly declines the performance as shown in 1

3 Affordance Prediction

3.1 Architecture

I used Resnet18 backbone along with MLPs for both conditional and unconditional modules. conditional modules are lane distance and route angle and they are conditioned on the command (left, right etc.). The unconditonal modules are traffic light distance and traffic light state with the latter being 0 or 1 depending on the state. I tried the same setup twice with a difference. Firstly, for the traffic light state, similar to the paper, I used an MLP with 2 output neurons and cross entropy loss. And for the rest I used L1 loss. Secondly, I tested with a single output layer and binary cross entorpy loss.

3.2 Training

I trained both models for 10 and 30 epochs with batch size of 120 and learning rate of 2e-4, Adam optimizer with weight decay of 1e-4. As for the loss function for the first architecuture I used L1 with cross entropy loss for the traffic light

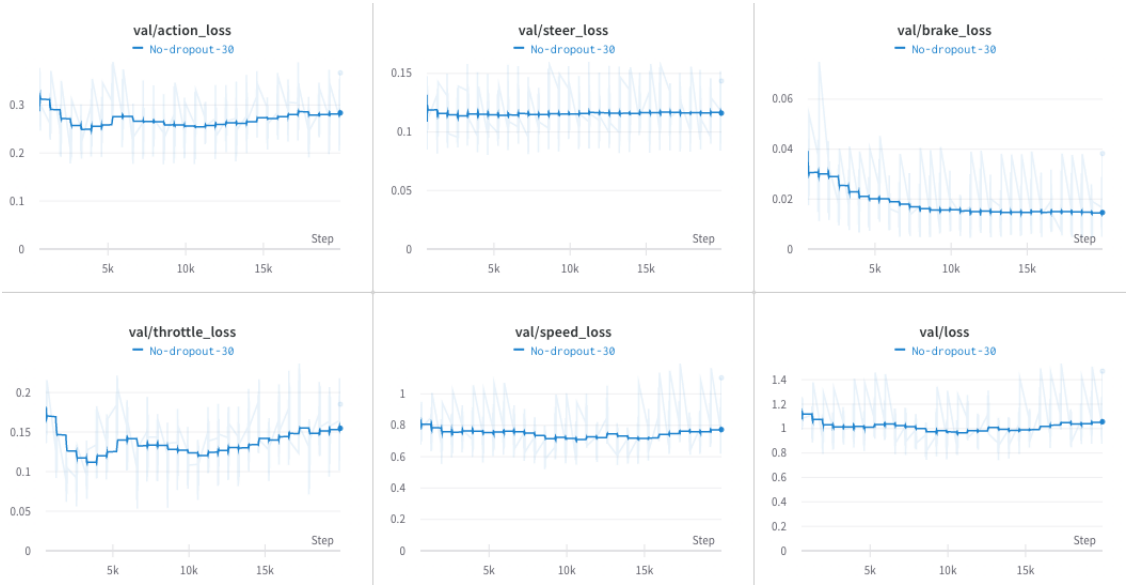


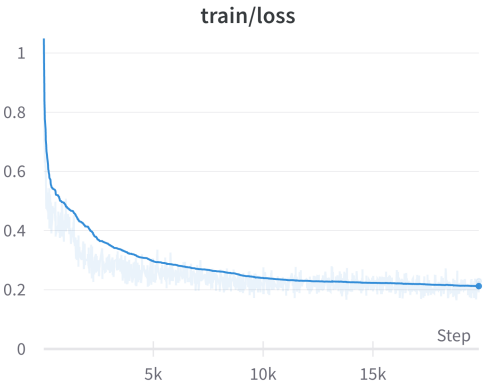
Figure 1: CILRS without dropout: slightly worsened results without dropout.

state and for the second model I used L1 but this time binary cross entropy loss.

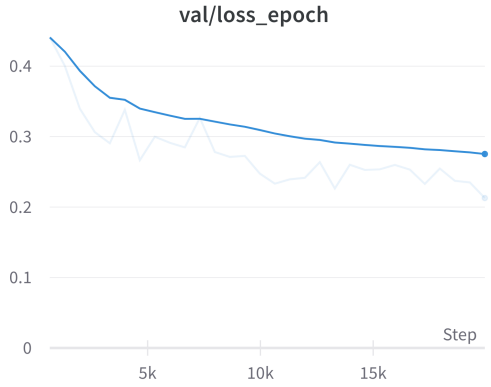
4 Results

For better visualization, I logged the losses for each module seperatly to visualize the effect of the changes I made more closely. The descriptions under images show the results.
[1]

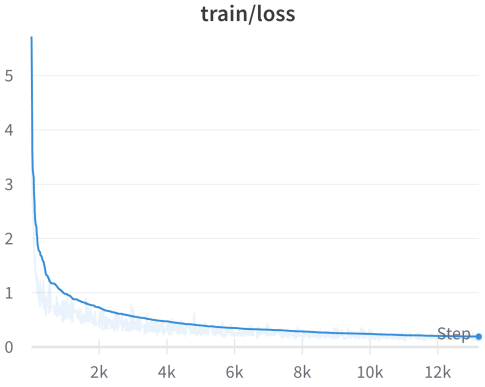
[1] I would like to note that I did everything I could for CARLA to work on a linux-based environment but failed. On clusters, I tried with newly build environment and even though I followed the instrucitons carefully, I got a lot of conflicts for libpng library. I will still be working on getting it work. Also, I tried these way before the deadline (Wenesday April 19th) :). If I get it working I will post the results on Github. Thank you for your time.



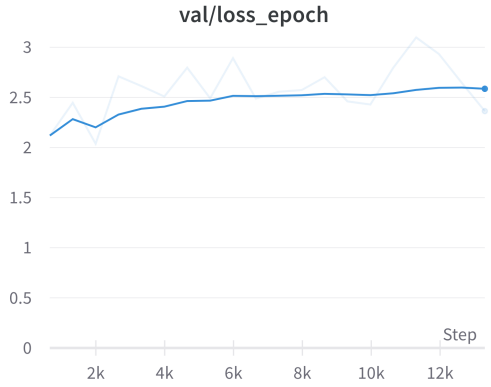
(a) CILRS: The training loss for L1. The loss started declining over time and as suggested also in the paper, the results were better with this loss.



(b) CILRS:The validation loss started decreasing over time for L1 loss and less overfitting could be seen. This was also due to the fact that I added dropout of 50% since the model extremely overfitted to steer and speed loss during training.



(c) CILRS: The training loss converges and decreases over time when using **MSE/L2** loss, however, the same does not apply to the validation loss



(d) CILRS: The validation loss does not decrease over time. The results were not good for MSE loss.

Figure 2: Comparison of training and validation loss for **L1 loss** (top row) and **MSE loss** (bottom row) for 30 epochs and batch size of 128.

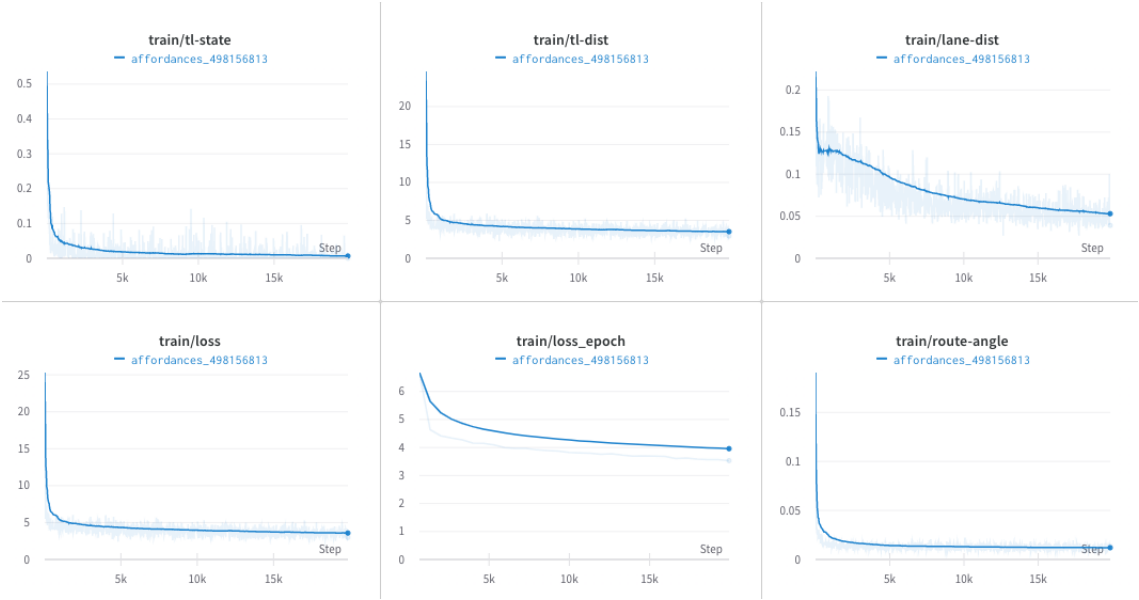


Figure 3: Affordances: The training loss for affordances. Here I'm using L1 loss and the aformentioned first architecture. Here, losses are plotted for each sub module. Train/loss refers to the overal loss.

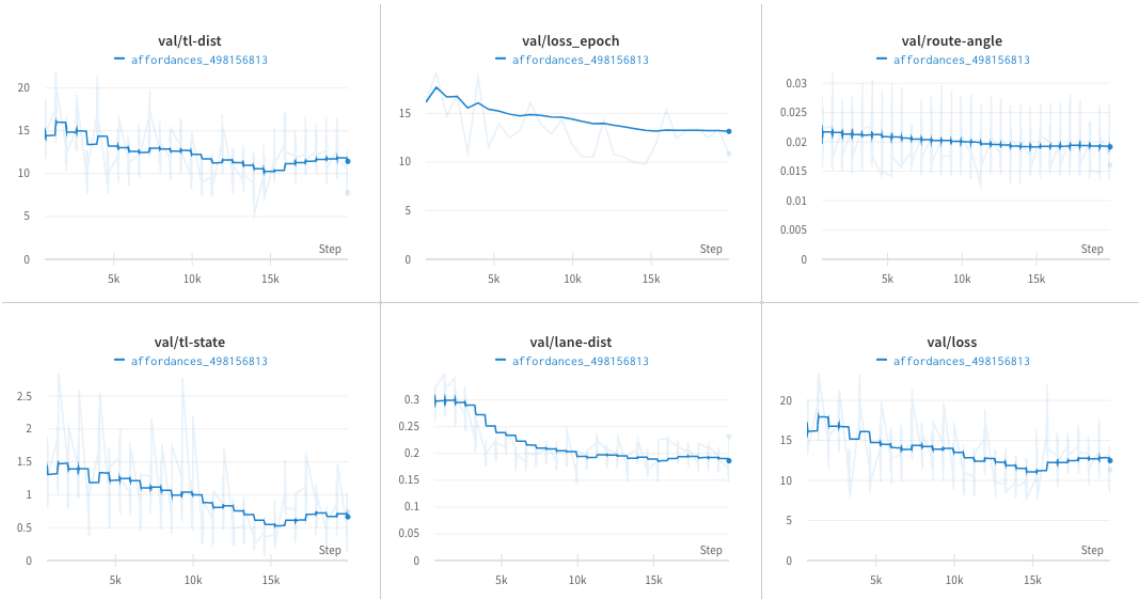
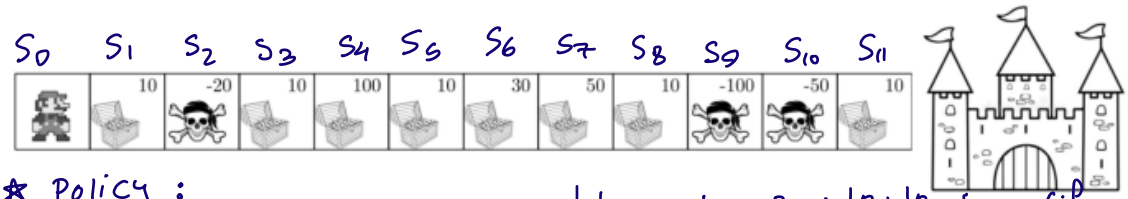


Figure 4: Affordances: The validation loss for affordances under the first architecture. Here loss_epoch refers to the overal loss.



★ Policy :

$a_0 = \hat{\pi}(S_0)$ => 3 steps => S_3

$a_1 = \hat{\pi}(S_1)$ = 3 steps => S_4

$a_2 = \hat{\pi}(S_2)$ = 2 steps => S_4

$a_3 = \hat{\pi}(S_3)$ => 3 steps => S_6

$a_4 = \hat{\pi}(S_4)$ => 3 steps => S_7

$a_5 = \hat{\pi}(S_5)$ => 2 steps => S_7

$a_6 = \hat{\pi}(S_6)$ => 2 steps => S_8

$a_7 = \hat{\pi}(S_7)$ => 3 steps => S_{10}

$a_8 = \hat{\pi}(S_8)$ => 3 steps => S_{11}

$a_9 = \hat{\pi}(S_9)$ => 2 steps => S_{11}

$a_{10} = \hat{\pi}(S_{10})$ => finish castle

$a_{11} = \hat{\pi}(S_{11})$ => finished castle

$V_{S_0} = 10 + 30 + 10 + 10 = 60$

$V_{S_1} = 10 + V_{S_4} = 110$

$V_{S_2} = -20 + V_{S_4} = 80$

$V_{S_3} = 10 + V_{S_6} = 60$

$V_{S_4} = 100 + V_{S_7} = 100$

$V_{S_5} = 10 + V_{S_7} = 10$

$V_{S_6} = 30 + V_{S_8} = 30 + 20 = 50$

$V_{S_7} = 50 + V_{S_{10}} = 0$ (-50)

$V_{S_8} = 10 + V_{S_{11}} = 20$

$V_{S_9} = -100 + V_{S_{11}} = -90$

$V_{S_{10}} = -50$

$V_{S_{11}} = 10$ $\quad V(\text{castle}) = 0$

(If we count mario's initial money (100) we can add the sum by 100).

$Q(S_t) \leftarrow Q(S_t) + \alpha(G_t^{(n)} - Q(S_t))$

$G_t^{(n)} = R_t + \gamma^n Q(S_{t+n})$

all actions: $\mathcal{A}$ $\quad$ A : two steps

B : three steps

$Q(S_0, A) = Q(S_0, A)^* + \alpha(\overbrace{R(A)}^{-20} + \max_{a \in \mathcal{A}} Q(S_1, a') - Q(S_0, A)^*$

$Q(S_1, A) = Q(S_1, A)^* + \alpha(\overbrace{R(A)}^{10} + \max_{a \in \mathcal{A}} Q(S_2, a') - Q(S_0, A)^*$

$Q(S_2, A) = Q(S_2, A)^* + \alpha(R($

at S_0 take action A $\rightarrow r_t = -20$

$$Q(S_0, A) = -20 + 1 \cdot \max_{a'} (Q(S_1, a')) - Q(S_0, a_0) = 80$$

at S_1 take action A $\rightarrow r_t = 10$

$$Q(S_1, A) = 10 + \max_{a'} (Q(S_2, a')) - Q(S_1, a_1) = -20$$

at S_2 action A $\rightarrow r_t = 100$

$$Q(S_2, A) = 10 + \max_{a'} (Q(S_3, a') - Q(S_2, a_2) = -10$$

at S_3 action A $\rightarrow r_t = 10$

$$Q(S_3, A) = 10 + \max(Q_4, a') - Q(S_{max}, A) = 110 - 60 = 50$$

$Q(S_4, A) = 100 + 10 - 100 = 10$

$Q(S_5, A) = 10 + 50 - 10 = 50$

$Q(S_6, A) = 30 + 0 - 50 = -20$

$Q(S_7, A) = 50 + 20 = 70$

$Q(S_8, A) = 10 - 100 + 20 = -70$

$Q(S_9, A) = 50 - 50 - 100 = -100$

$Q(S_{10}, A) = -50 + 10 - 50 = 20$

$Q(S_{11}, A) = 10 + 0 - 10 = 0$