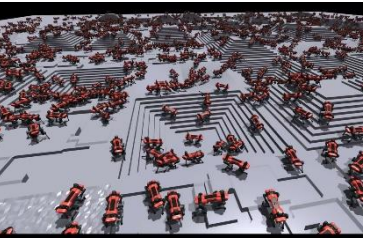
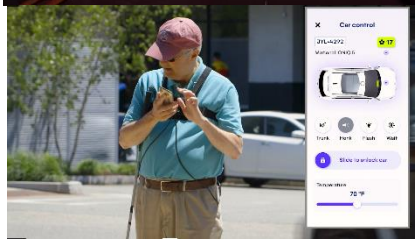
EC500: Robot Learning and Vision for Navigation



Eshed Ohn-Bar

January 30, 2023



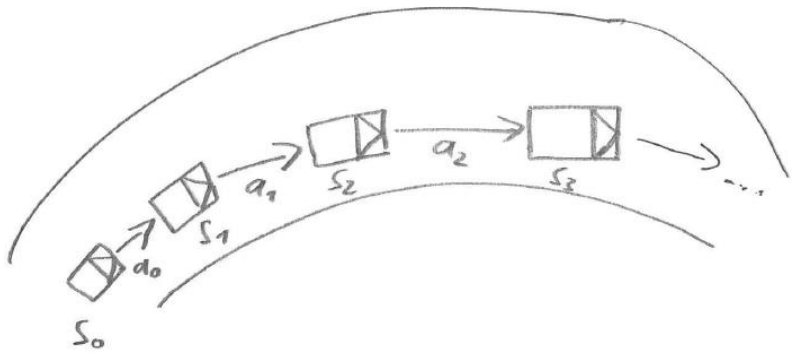
Issue of Domain/Covariate Shift

- Humans can do things that robots can’t do (and vice versa)

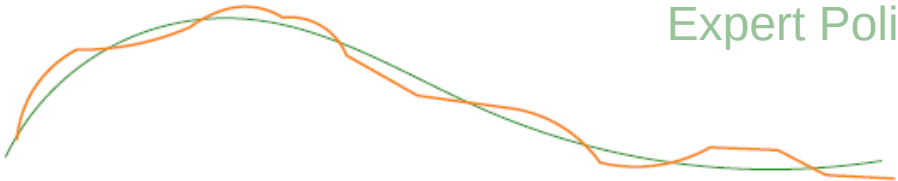
- Humans understand differently from robots

Problem: Actions are interdependent along the trajectory!

But, behavior cloning only learns/predicts one step at a time?
(that is, trained over pairs of (s_t, a_t))



Errors Independent in Time



Error at time t with probability ε

$E[\text{Total errors}] \lesssim \varepsilon T$

Compounding Errors

Neural Net Policy

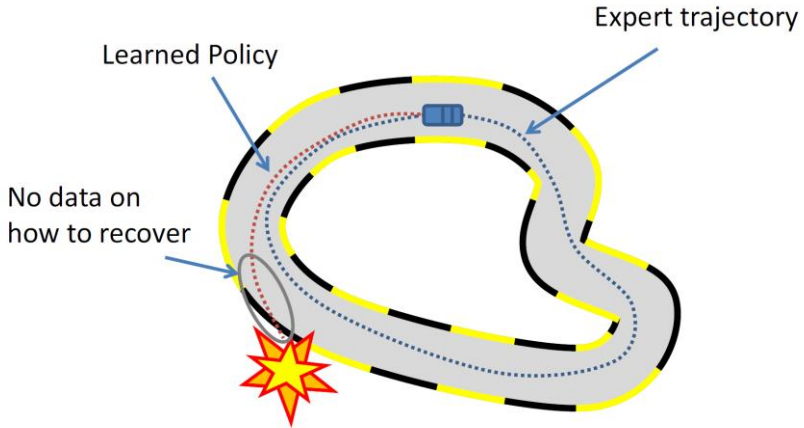
Expert Policy

error at time t with probability ε

$E[\text{Total errors}] \lesssim \varepsilon(T + (T-1) + (T-2) + \ldots + 1) \propto \varepsilon T^2$

Data Distribution Mismatch

$$P(s|\pi_\theta) \; != \; P(s|\pi^*)$$



Formal Definition of Imitation Learning

General Imitation Learning:

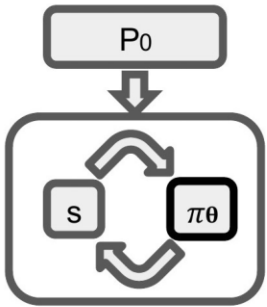
$$\underset{\theta}{\operatorname{argmin}}\ \mathbb{E}_{s\sim P(s|\pi_\theta)}\left[\mathcal{L}\left(\pi^*(s), \pi_\theta(s)\right)\right]$$

- State distribution $P(s|\pi_\theta)$ depends on rollout determined by current policy π_θ

Behavior Cloning:

$$\underset{\theta}{\operatorname{argmin}}\ \underbrace{\mathbb{E}_{(s^*,a^*)\sim P^*}\left[\mathcal{L}\left(a^*, \pi_\theta(s^*)\right)\right]}_{=\sum_{i=1}^{N}\mathcal{L}\left(a_i^*, \pi_\theta(s_i^*)\right)}$$

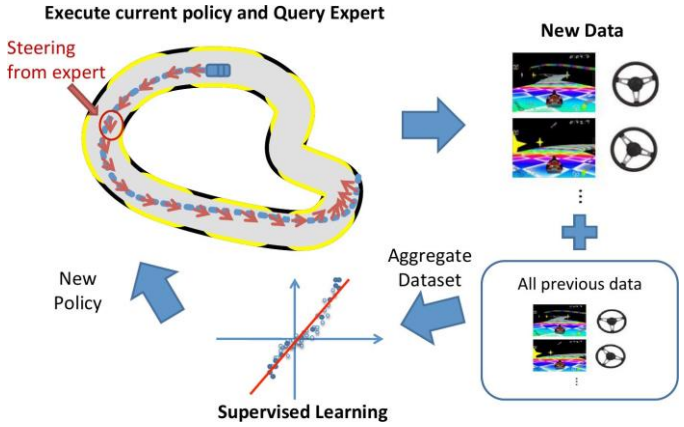
- State distribution P^* provided by expert
- Reduces to supervised learning problem



Challenges of Behavior Cloning

- Behavior cloning reasons only about immediate next step
- Behavior cloning makes IID assumption
  - Next state is sampled from states observed during expert demonstration
  - Thus, next state is sampled independently from action predicted by current policy
- What if π_θ makes a mistake?
  - Enters new states that haven't been observed before
  - New states not sampled from same (expert) distribution anymore
  - Cannot recover, catastrophic failure in the worst case
- What can we do to overcome this train/test distribution mismatch?

DAGGER: Dataset Aggregation



Idea: Iteratively build a set of inputs that the final policy is likely to encounter based on previous experience. Query expert for aggregate dataset.

Ross, Gordon and Bagnell: A Reduction of Imitation Learning and Structured Prediction to No-Regret Online Learning. AISTATS, 2011.

DAGGER: Dataset Aggregation

Initialize $\mathcal{D} \leftarrow \emptyset$.
Initialize $\hat{\pi}_1$ to any policy in Π.
for $i = 1$ **to** N **do**
  Let $\pi_i = \beta_i \pi^* + (1 - \beta_i)\hat{\pi}_i$.
  Sample T-step trajectories using π_i.
  Get dataset $\mathcal{D}_i = \{(s, \pi^*(s))\}$ of visited states by π_i and actions given by expert.
  Aggregate datasets: $\mathcal{D} \leftarrow \mathcal{D} \bigcup \mathcal{D}_i$.
  Train classifier $\hat{\pi}_{i+1}$ on $\mathcal{D}$.
end for
Return best $\hat{\pi}_i$ on validation.

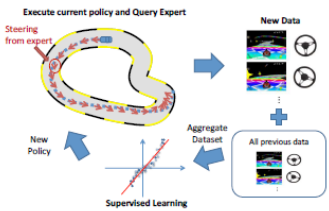
- Let T be the number of rollout time steps and ϵ be the probability of a mistake
- Behavior cloning: $O(T^2\epsilon)$ regret, DAGGER: no regret (in suitable conditions)
  - Regret = loss of current policy - loss of perfect policy
- But requires to flexibly run the environment and query the expert during training

Ross, Gordon and Bagnell: A Reduction of Imitation Learning and Structured Prediction to No-Regret Online Learning. AISTATS, 2011.

DAGGER: In Simulation

Dataset AGGregation: bring learner's and expert's trajectory distributions closer by labelling additional data points resulting from applying the current policy

1. train $\pi_\theta(u_t|o_t)$ from human data $\mathcal{D}_{\pi^*} = \{o_1, u_1, ..., o_N, u_N\}$
2. run $\pi_\theta(u_t|o_t)$ to get dataset $\mathcal{D}_\pi = \{o_1, ..., o_M\}$
3. Ask human to label $\mathcal{D}_\pi$ with actions u_t
4. Aggregate: $\mathcal{D}_{\pi^*} \leftarrow \mathcal{D}_{\pi^*} \cup \mathcal{D}_\pi$
5. GOTO step 1.



Problems:

- execute an unsafe/partially trained policy
- repeatedly query the expert

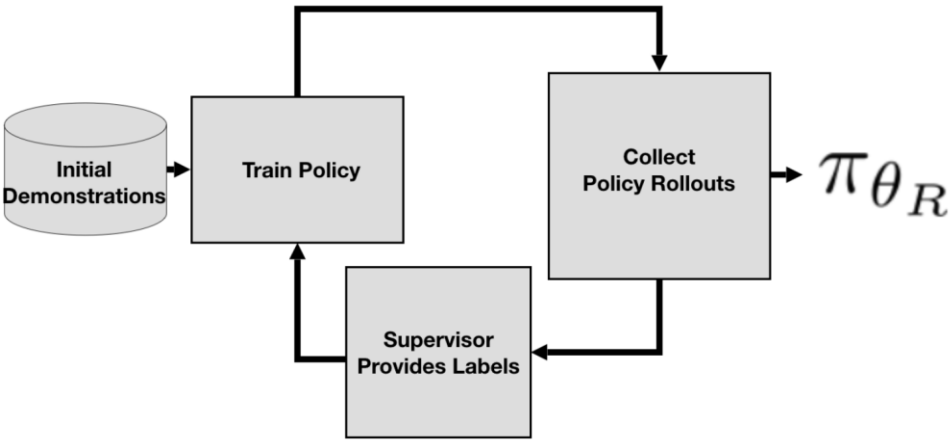
Control Notation

o - observation

u - action

Ross, Gordon and Bagnell: A Reduction of Imitation Learning and Structured Prediction to No-Regret Online Learning. AISTATS, 2011.

Reducing Shift with On-Policy Methods

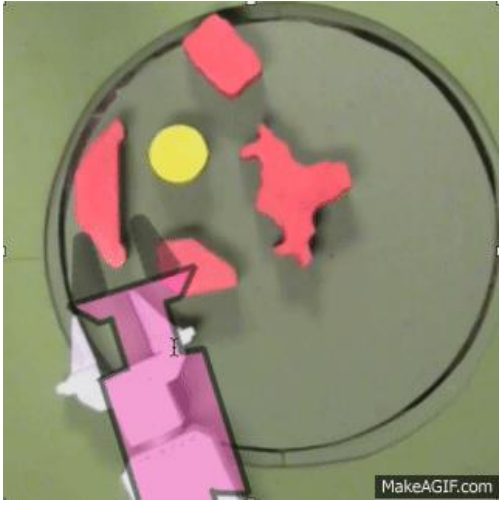


The DAgger Algorithm.

DAGGER: In Real Platform

- Difficult for the expert , reaction time of human (e.g., drone, could cause suboptimal actions)
- Safe? Executing an imperfect policy.
- Computationally intensive

Difficulty of Getting Feedback from Expert



DAGGER: In Real Platform

Application on drones: given RGB from the drone camera predict steering angles

Learning monocular reactive UAV control in cluttered natural environments, Ross et al. 2013

Fig. 2. One frame from MAV camera stream. The white line indicates the current yaw commanded by the current DAgger policy π_{n-1} while the red line indicates the expert commanded yaw. In this frame DAgger is wrongly heading for the tree in the middle while the expert is providing the correct yaw command to go to the right instead. These expert controls are recorded for training later iterations but not executed in the current run.

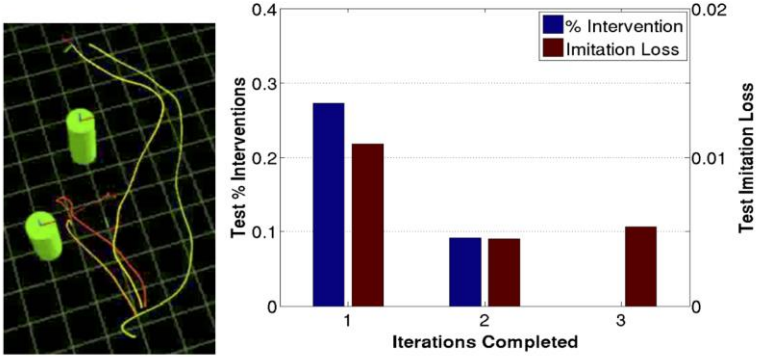
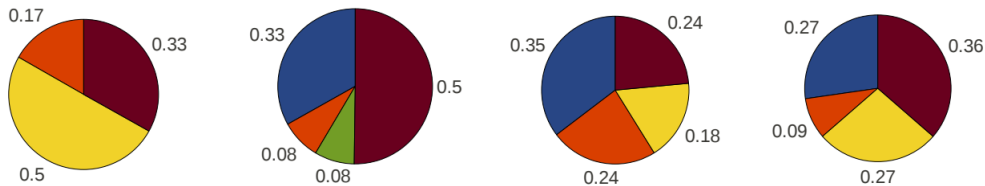


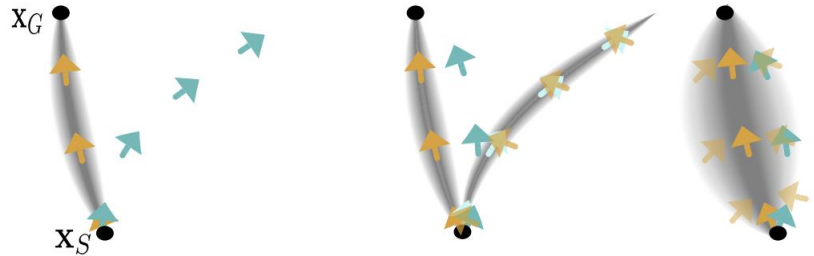
Fig. 4. Left: Improvement of trajectory by DAgger over the iterations. The rightmost green trajectory is the pilot demonstration. The short trajectories in red & orange show the controller learnt in the 1^{st} and 2^{nd} iterations which fail. The 3^{rd} iteration controller succesfully avoids both obstacles and is similar to the demonstrated trajectory. Right: Percentage of scenarios the pilot had to intervene and the imitation loss (average squared error in controls of controller to human expert on hold-out data) after each iteration of Dagger. After 3 iterations, there was no need for the pilot to intervene and the UAV could successfully avoid all obstacles



(a) Pilot (b) After iter. 1 (c) After iter. 2 (d) After iter. 3

Fig. 8. Percentage of failures of each type for DAgger over the iterations of training in the high-density region. Blue: Large Trees, Orange: Thin Trees, Yellow: Leaves and Branches, Green: Other obstacles (poles, signs, etc.), Red: Too Narrow FOV. Clearly, a majority of the crashes happen due to a too narrow FOV and obstacles which are hard to perceive, such as branches and leaves.

Off-Policy Imitation Learning with Noise Injection



$$\pi_{\theta^*}(\mathbf{u}|\mathbf{x}, \psi) = \mathcal{N}(\pi_{\theta^*}(\mathbf{x}), \Sigma)$$

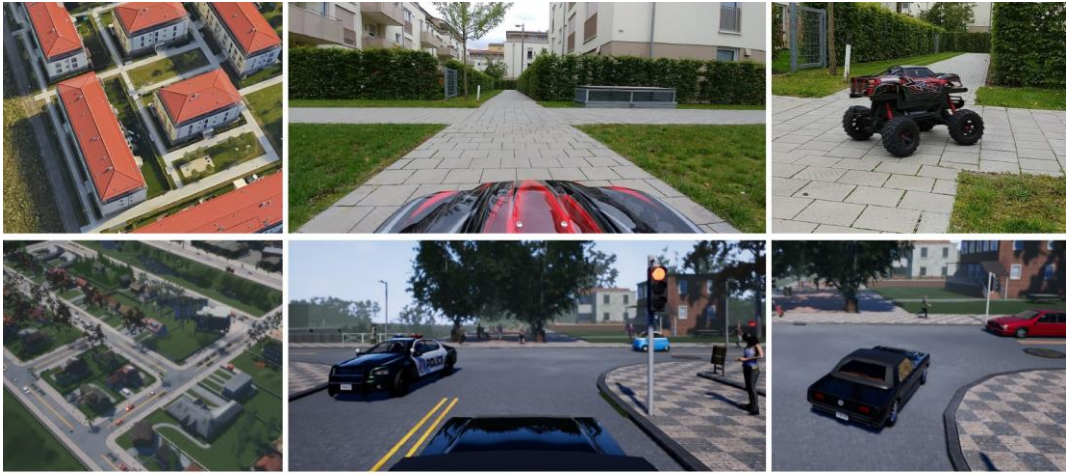
Laskey et al., DART: Noise Injection for Robust Imitation Learning, 2017

Laskey et al., DART: Noise Injection for Robust Imitation Learning, 2017

When else does
behavior cloning fail?

Conditional Imitation Learning

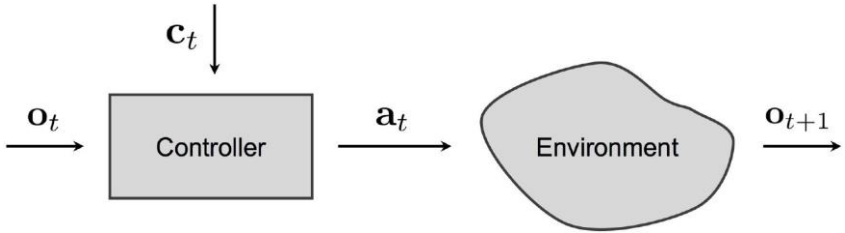
Conditional Imitation Learning



(a) Aerial view of test environment (b) Vision-based driving, view from onboard camera (c) Side view of vehicle

Codevilla, Mu ̈ller, Lo ́pez, Koltun and Dosovitskiy: End-to-End Driving Via Conditional Imitation Learning. ICRA, 2018.

Conditional Imitation Learning



Idea:

- Condition controller on **navigation command** $c \in \{left, right, straight\}$
- High-level navigation command can be provided by consumer GPS, i.e., telling the vehicle to **turn left/right** or go **straight** at the next intersection
- This removes the task ambiguity induced by the environment (not: noise)
- Observation o_t: current image Action a_t: steering angle & acceleration

Codevilla, Müller, López, Koltun and Dosovitskiy: End-to-End Driving Via Conditional Imitation Learning. ICRA, 2018.

Comparison to Behavior Cloning

BC

- Training Set:

$$\mathcal{D} = \{(a_i^*, s_i^*)_{i=1}^N\}$$

- Objective:

$$\underset{\theta}{\operatorname{argmin}} \sum_{i=1}^{N} \mathcal{L}\left(a_i^*, \pi_\theta(s_i^*)\right)$$

- Assumption:

$$\exists f(\cdot) : a_i = f(s_i)$$

Often violated in practice! Why?

Conditional BC

- Training Set:

$$\mathcal{D} = \{(a_i^*, s_i^*, c_i^*)_{i=1}^N\}$$

- Objective:

$$\underset{\theta}{\operatorname{argmin}} \sum_{i=1}^{N} \mathcal{L}\left(a_i^*, \pi_\theta(s_i^*, c_i^*)\right)$$

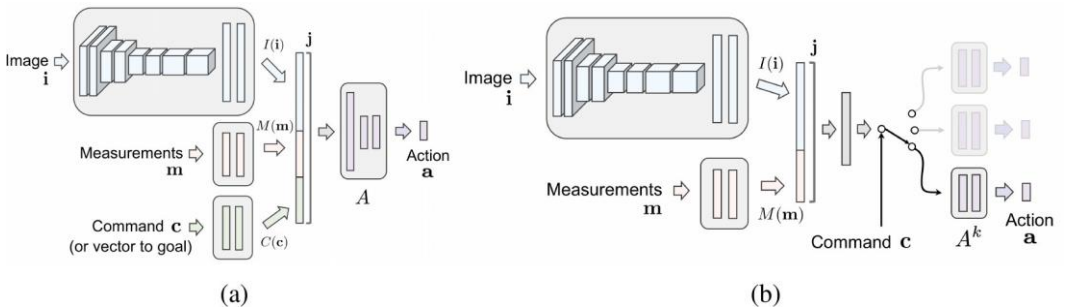
- Assumption:

$$\exists f(\cdot, \cdot) : a_i = f(s_i, c_i)$$

Better assumption!

Codevilla, Muller, Lopez, Koltun and Dosovitskiy: End-to-End Driving Via Conditional Imitation Learning. ICRA, 2018.

Conditional Imitation Learning: Network Architecture

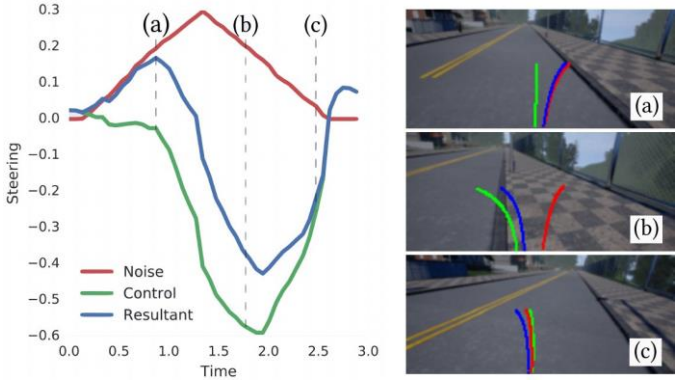


- This paper proposes two network architectures:
  - (a) Extract features $C(c)$ and concatenate with image features $I(\mathbf{i})$
  - (b) Command c acts as switch between specialized submodules
- Measurements m capture additional information (here: speed of vehicle)

Codevilla, Muller, Lopez, Koltun and Dosovitskiy: End-to-End Driving Via Conditional Imitation Learning. ICRA, 2018.

Does CIL also suffer from catastrophic drift?

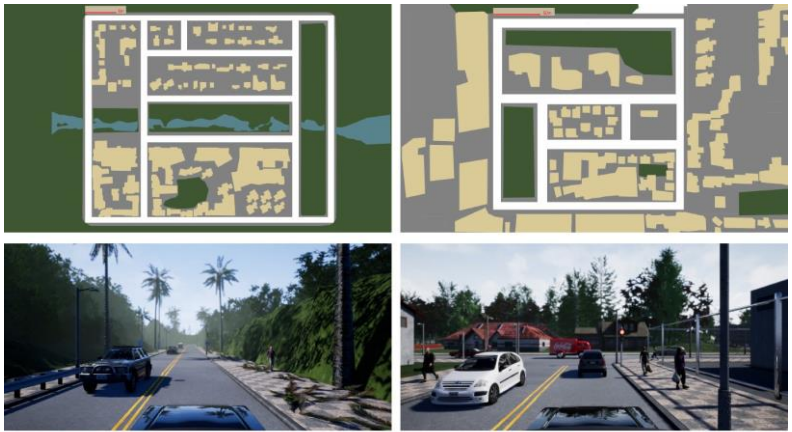
Conditional Imitation Learning: Noise Injection



- Temporally correlated noise injected into trajectories ⇒ drift
- Record driver's (=expert's) corrective response ⇒ recover from drift

Codevilla, Muller, Lopez, Koltun and Dosovitskiy: End-to-End Driving Via Conditional Imitation Learning. ICRA, 2018.

CARLA Simulator



Town 1 (training) Town 2 (testing)

http://www.carla.org

Codevilla, Muller, Lopez, Koltun and Dosovitskiy: End-to-End Driving Via Conditional Imitation Learning. ICRA, 2018.

Conditional Imitation Learning: Results

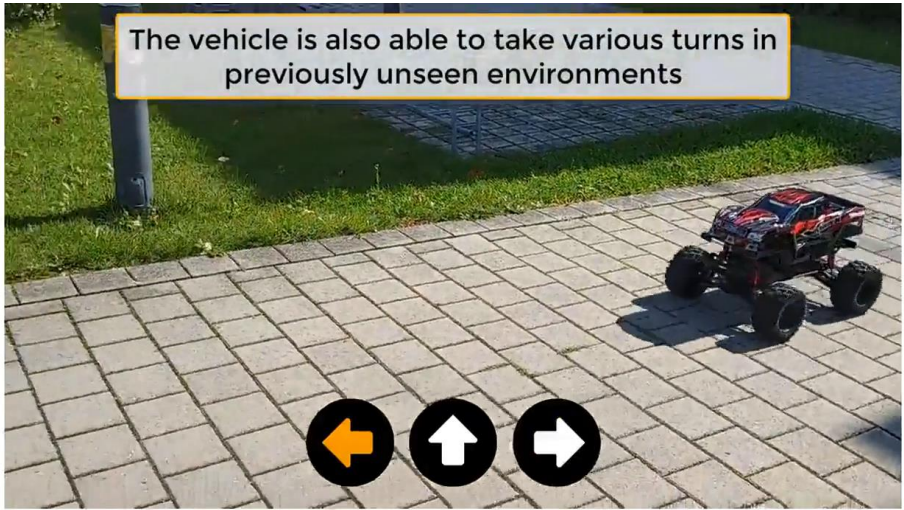
Model	Success rate		Km per infraction	
	Town 1	Town 2	Town 1	Town 2
Non-conditional	20%	26%	5.76	0.89
Goal-conditional	24%	30%	1.87	1.22
Ours branched	**88%**	**64%**	2.34	1.18
Ours cmd. input	78%	52%	3.97	1.30
Ours no noise	56%	22%	1.31	0.54
Ours no aug.	80%	0%	4.03	0.36
Ours shallow net	46%	14%	0.96	0.42

Methods:

- Goal-conditional: provided with vector to goal (i.e., compass)
- Ours branched: Network architecture (b)
- Ours cmd. input: Network architecture (a)
- Ours no noise: no noise injected into trajectories (see previous slide)
- Ours no aug.: no augmentation (contrast, brightness, blur, noise, region dropout)

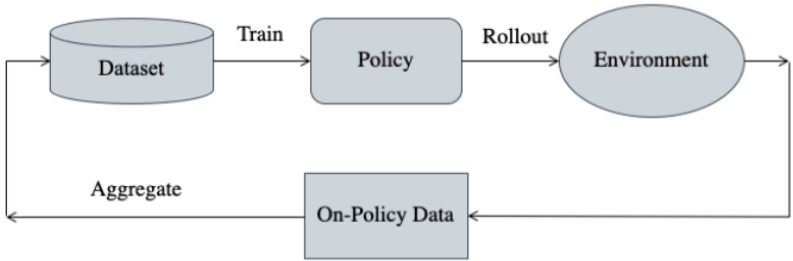
Codevilla, Muller, Lopez, Koltun and Dosovitskiy: End-to-End Driving Via Conditional Imitation Learning. ICRA, 2018.

Conditional Imitation Learning: Results



 Wed PM Pod M.4

DAgger



Data Aggregation (DAgger):

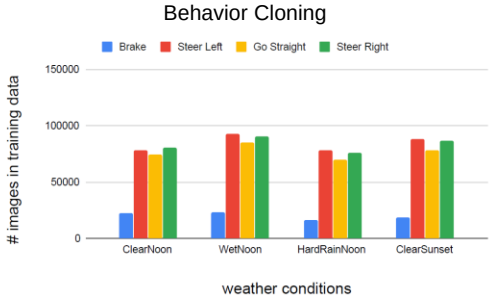
- Iteratively build a set of inputs that the final policy is likely to encounter based on previous experience. Query expert for aggregate dataset.
- But can easily overfit to main mode of demonstrations

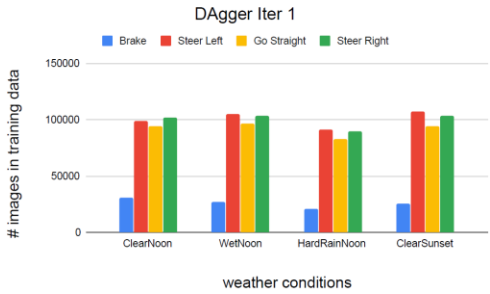

60

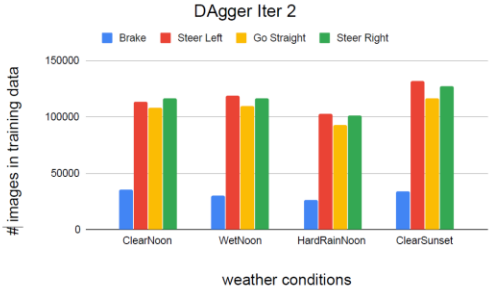

30

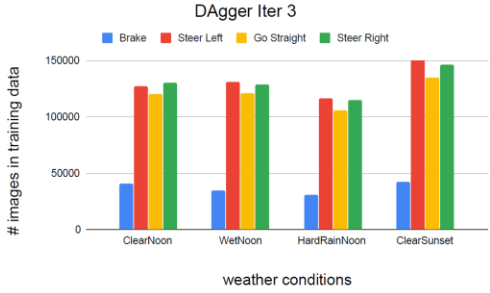
Common Failures for BC

Collision with Pedestrian


Behavior Cloning
Brake
Steer Left
Go Straight
Steer Right
150000
100000
50000
0
images in training data
ClearNoon
WetNoon
HardRainNoon
ClearSunset
weather conditions


DAgger Iter 1
Brake
Steer Left
Go Straight
Steer Right
150000
100000
50000
0
images in training data
ClearNoon
WetNoon
HardRainNoon
ClearSunset
weather conditions


DAgger Iter 2
Brake
Steer Left
Go Straight
Steer Right
150000
100000
50000
0
images in training data
ClearNoon
WetNoon
HardRainNoon
ClearSunset
weather conditions


DAgger Iter 3
Brake
Steer Left
Go Straight
Steer Right
150000
100000
50000
0
images in training data
ClearNoon
WetNoon
HardRainNoon
ClearSunset
weather conditions

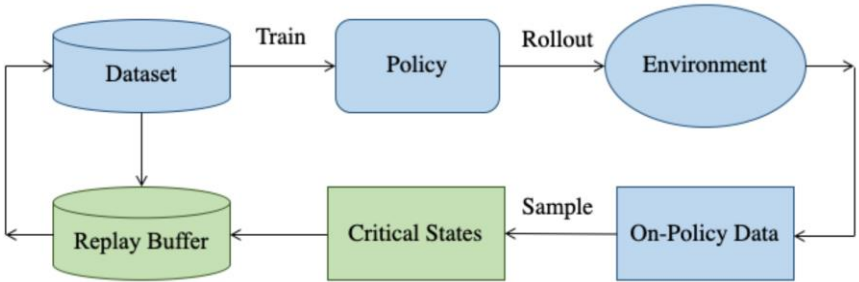
DAGGER: Dataset Aggregation

Initialize $\mathcal{D} \leftarrow \emptyset$.
Initialize $\hat{\pi}_1$ to any policy in Π.
for $i = 1$ **to** N **do**
  Let $\pi_i = \beta_i \pi^* + (1 - \beta_i)\hat{\pi}_i$.
  Sample T-step trajectories using π_i.
  Get dataset $\mathcal{D}_i = \{(s, \pi^*(s))\}$ of visited states by π_i and actions given by expert.
  Aggregate datasets: $\mathcal{D} \leftarrow \mathcal{D} \bigcup \mathcal{D}_i$.
  Train classifier $\hat{\pi}_{i+1}$ on $\mathcal{D}$.
end for
Return best $\hat{\pi}_i$ on validation.

- Let T be the number of rollout time steps and ϵ be the probability of a mistake
- Behavior cloning: $O(T^2\epsilon)$ regret, DAGGER: no regret (in suitable conditions)
  - Regret = loss of current policy - loss of perfect policy
- But requires to flexibly run the environment and query the expert during training

Ross, Gordon and Bagnell: A Reduction of Imitation Learning and Structured Prediction to No-Regret Online Learning. AISTATS, 2011.

DAGGER with Critical States and Replay Buffer



Key Ideas:

1. Sample **critical states** from the collected on-policy data based on the utility they provide to the learned policy in terms of driving behavior
2. Incorporate a **replay buffer** which progressively focuses on the high uncertainty regions of the policy's state distribution

Exploring Data Aggregation in Policy Learning for Vision-based Urban Autonomous Driving. CVPR, 2020.

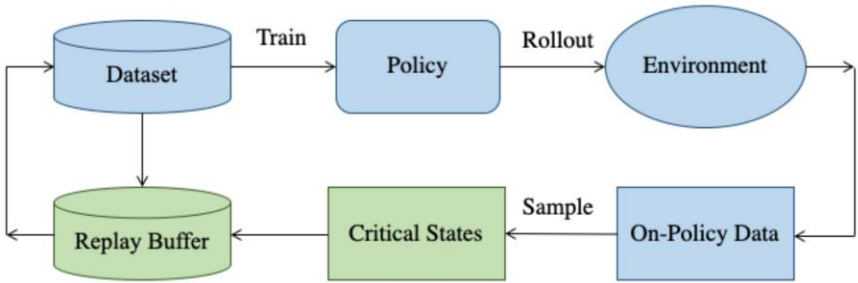
Algorithm 1 DAgger with Critical States and Replay Buffer

Collect D_0 using expert policy π^*
$\hat{\pi}_0 = \text{argmin}_{\pi} \mathcal{L}(\pi, \pi^*, D_0)$
Initialize replay buffer $D \leftarrow D_0$
Let $m = |D_0|$
for i = 1 *to* N **do**
    Generate on-policy trajectories using $\hat{\pi}_{i-1}$
    Get dataset $D_i = \{(s, \pi^*(s))\}$ of visited states by $\hat{\pi}_{i-1}$ and actions given by expert
    Get $D'_i \leftarrow \{(s_c, \pi^*(s_c))\}$ after sampling critical states from D_i
    Combine datasets: $D \leftarrow D \cup D'_i$
    while $|D| > m$ **do**
        Sample $(s, \pi^*(s))$ randomly from $D \cap D_0$
        $D \leftarrow D - \{(s, \pi^*(s))\}$
    end
    Train $\hat{\pi}_i = \text{argmin}_{\pi} \mathcal{L}(\pi, \pi^*, D)$ with policy initialized from $\hat{\pi}_{i-1}$
end
return $\hat{\pi}_N$

Algorithm 1 DAgger with Critical States and Replay Buffer

Collect D_0 using expert policy π^*
$\hat{\pi}_0 = \text{argmin}_{\pi} \mathcal{L}(\pi, \pi^*, D_0)$
Initialize replay buffer $D \leftarrow D_0$
Let $m = |D_0|$
for i = 1 *to* N **do**
    Generate on-policy trajectories using $\hat{\pi}_{i-1}$
    Get dataset $D_i = \{(s, \pi^*(s))\}$ of visited states by $\hat{\pi}_{i-1}$ and actions given by expert
    Get $D'_i \leftarrow \{(s_c, \pi^*(s_c))\}$ after sampling critical states from D_i
    Combine datasets: $D \leftarrow D \cup D'_i$
    while $|D| > m$ **do**
        Sample $(s, \pi^*(s))$ randomly from $D \cap D_0$
        $D \leftarrow D - \{(s, \pi^*(s))\}$
    end
    Train $\hat{\pi}_i = \text{argmin}_{\pi} \mathcal{L}(\pi, \pi^*, D)$ with policy initialized from $\hat{\pi}_{i-1}$
end
return $\hat{\pi}_N$

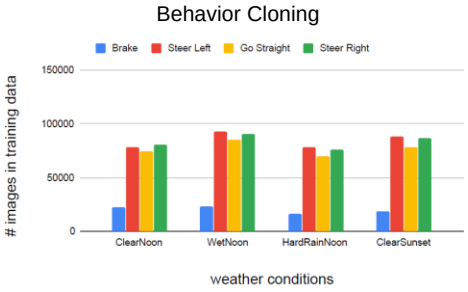
DAGGER with Critical States and Replay Buffer

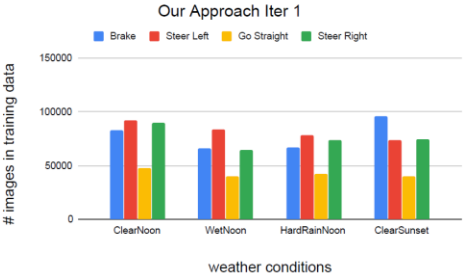


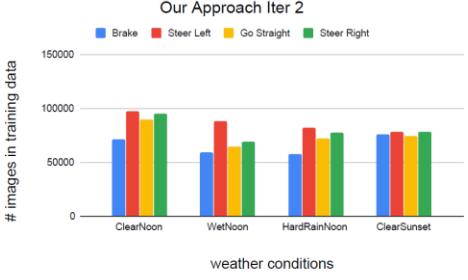
Sampling Strategies:

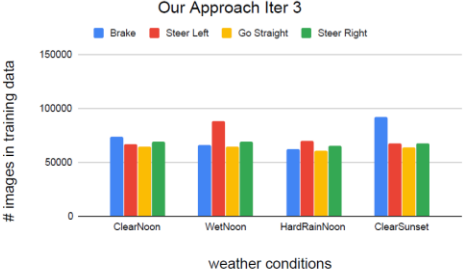
- Task-based: Sample uniformly from “left”, “right”, “straight”
- Policy-based: Use test-time dropout to estimate epistemic uncertainty
- Expert-based: Highest loss or deviation in brake signal wrt. expert

Task	CILRS	DART	DA-RB (Ours)	Expert
Training	45±6	50±1	**66**±5	71±4
New Weather	39±4	37±2	56±1	72±3
New Town	23±1	26±2	**36**±3	41±2
New Town & Weather	26±2	21±1	**35**±2	43±2


Behavior Cloning
Brake
Steer Left
Go Straight
Steer Right
150000
100000
50000
0
images in training data
ClearNoon
WetNoon
HardRainNoon
ClearSunset
weather conditions


Our Approach Iter 1
Brake
Steer Left
Go Straight
Steer Right
150000
100000
50000
0
images in training data
ClearNoon
WetNoon
HardRainNoon
ClearSunset
weather conditions


Our Approach Iter 2
Brake
Steer Left
Go Straight
Steer Right
150000
100000
50000
0
images in training data
ClearNoon
WetNoon
HardRainNoon
ClearSunset
weather conditions


Our Approach Iter 3
Brake
Steer Left
Go Straight
Steer Right
150000
100000
50000
0
images in training data
ClearNoon
WetNoon
HardRainNoon
ClearSunset
weather conditions

Interpretability: GradCAM Attention Maps

CILRS [Codevilla et al. 2019]

Our Approach

Prakash et al., Exploring Data Aggregation in Policy Learning for Vision-based Urban Autonomous Driving, CVPR 2020

Interpretability: GradCAM Attention Maps

CILRS [Codevilla et al. 2019]

Our Approach

Prakash et al., Exploring Data Aggregation in Policy Learning for Vision-based Urban Autonomous Driving, CVPR 2020

CILRS+ (Codevilla et al. 2019)

DA-RB+ (Our Approach)

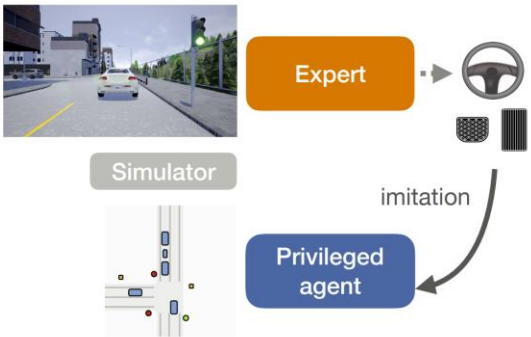
SafeDagger – Human takes over when safety policy determines they should

Algorithm 1 SafeDAgger Blue fonts are used to highlight the differences from the vanilla DAgger.

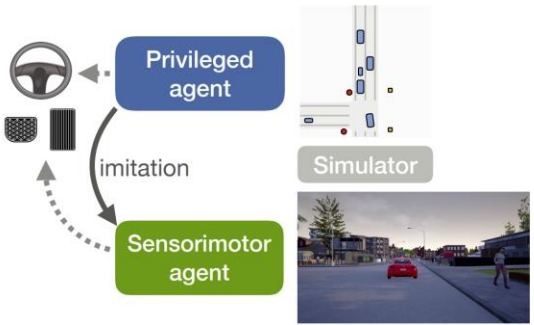
Collect D_0 using a reference policy π^*
Collect D_{safe} using a reference policy π^*
$\pi_0 = \arg\min_\pi l_{\text{supervised}}(\pi, \pi^*, D_0)$
$\pi_{\text{safe},0} = \arg\min_{\pi_{\text{safe}}} l_{\text{safe}}(\pi_{\text{safe}}, \pi_0, \pi^*, D_{\text{safe}} \cup D_0)$
for $i = 1$ **to** M **do**
  Collect D' using the **safety strategy** using π_{i-1} and $\pi_{\text{safe},i-1}$
  Subset Selection: $D' \leftarrow \{\phi(s) \in D' | \pi_{\text{safe},i-1}(\pi_{i-1}, \phi(s)) = 0\}$
  $D_i = D_{i-1} \cup D'$
  $\pi_i = \arg\min_\pi l_{\text{supervised}}(\pi, \pi^*, D_i)$
  $\pi_{\text{safe},i} = \arg\min_{\pi_{\text{safe}}} l_{\text{safe}}(\pi_{\text{safe}}, \pi_i, \pi^*, D_{\text{safe}} \cup D_i)$
end for
return π_M and $\pi_{\text{safe},M}$

Zhang and Cho, **Query-Efficient Imitation Learning for End-to-End Autonomous Driving,** AAAI 2017

Learning by Cheating, Chen et al., Conference on Robot Learning, 2019

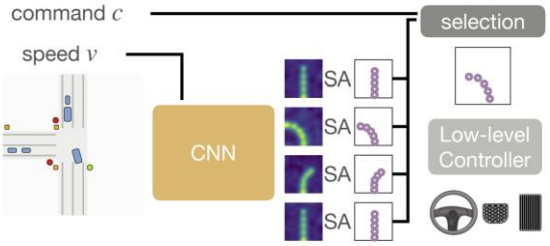


(a) Privileged agent imitates the expert

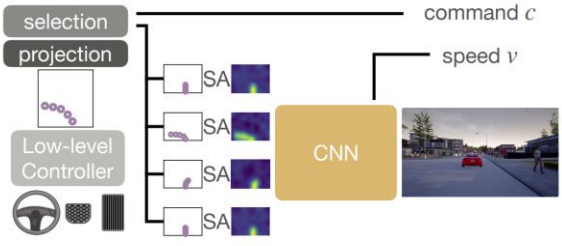


(b) Sensorimotor agent imitates the privileged agent

Learning by Cheating, Chen et al., Conference on Robot Learning, 2019

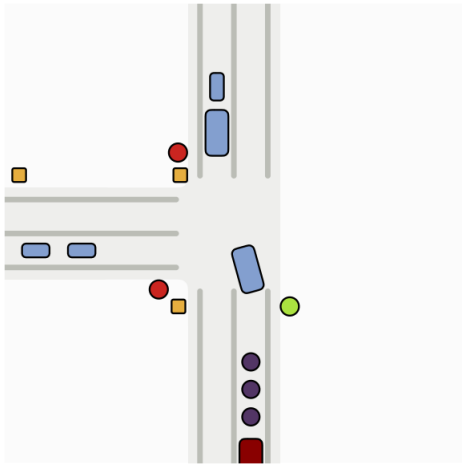


(a) Privileged agent

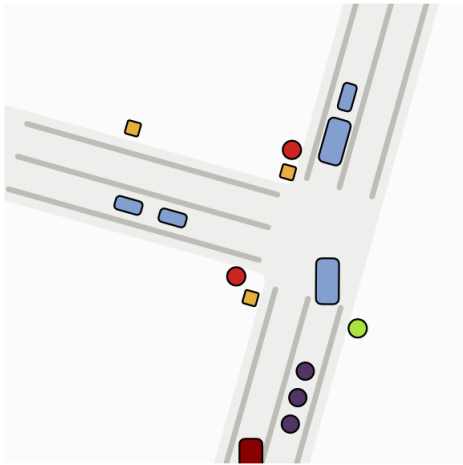


(b) Sensorimotor agent

Augmentation from the Expert



(a) Road map



(b) Rotation and shift aug.

Augmentation from the Expert

Supervision	white-box	on-policy	
Direct			20
Two stage			16
Two stage		✓	64
Two stage	✓		96
Two stage	✓	✓	100

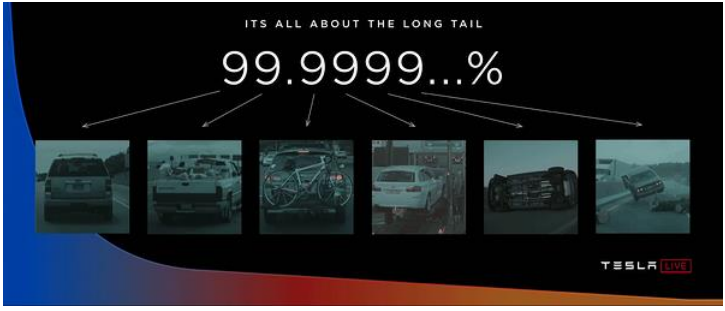
Table 1: Ablation study on the *CoRL2017* benchmark (CARLA 0.9.5, "navigation" condition, test town, test weather). Two key advantages of the presented decomposition – white-box supervision and on-policy trajectories – each substantially improve performance and together achieve 100% success rate on the benchmark.

Learning by Cheating

Task	Weather	CARLA ≤0.9.5 CIL[6]	CAL[20]	CILRS[7]	LBC	CARLA 0.9.6 LBC	PV	AT
Empty	train	48 ± 3	36 ± 6	51 ± 1	**100** ± 0	100 ± 0	100 ± 0	100 ± 0
Regular	train	27 ± 1	26 ± 2	44 ± 5	**96** ± 5	94 ± 3	95 ± 1	99 ± 1
Dense	train	10 ± 2	9 ± 1	38 ± 2	**89** ± 1	51 ± 3	46 ± 8	60 ± 3
Empty	test	24 ± 1	25 ± 3	90 ± 2	**100** ± 2	70 ± 0	100 ± 0	100 ± 0
Regular	test	13 ± 2	14 ± 2	87 ± 5	**94** ± 4	62 ± 2	93 ± 2	99 ± 1
Dense	test	2 ± 0	10 ± 0	67 ± 2	**85** ± 1	39 ± 8	45 ± 10	59 ± 6

Table 3: Comparison of the success rate of the presented approach (LBC) to the previous approaches on the *NoCrash* benchmark in the test town. (The supplement provides results on the training town.) PV denotes the performance of the privileged agent, AT is the performance of the built-in CARLA autopilot. Since the graphics and simulator behavior changed significantly with CARLA 0.9.6, we evaluate and compare our method on CARLA 0.9.5. CILRS was also run on this version of CARLA. Our approach outperforms prior work by significant factors, achieving 100% success rate in the "Empty" condition and reaching 85% success rate or higher in other conditions.

(Human Expert) Data is Essential



(Human Expert) Data is Essential

One Company to Rule Them All?

ars TECHNICA SUBSCRIBE SIGN IN

MEET THE NEW BOSSES —

Waymo CEO John Krafcik steps down

Waymo ordered "up to" 82,000 vehicles in 2018. Today, it has "well over 600."

TIMOTHY B. LEE - 4/3/2021, 8:30 AM

Waymo ordered “up to 82,000” vehicles in 2018.Today, it has “well over 600”

yahoo/finance

Tesla has 'key advantage' over other automakers, analyst says

Emily McCormick · Reporter
January 13, 2020

Forbes

Jul 3, 2020, 10:10am EDT | 63,113 views

Tesla: King Of Self-Driving Cars? Unbelievable

Trefis Team Contributor
Great Speculations Contributor Group
Markets

electrek

Exclusives Autos Alt. Transport Autonomy Energy Tesla Shop

OCTOBER 24, 2020

Tesla is collecting insane amount of data from its Full Self-Driving test fleet

Fred Lambert - Oct. 24th 2020 3:09 pm ET @FredericLambert

Underlying Development Process is Inefficient

Difficult for one entity to capture all modes (***unsafe***)
Bulk of data is kept private (***inaccessible, unsafe***)
Redundancy and high cost (***slow progress, urgent application***)

Underlying Development Process is Inefficient

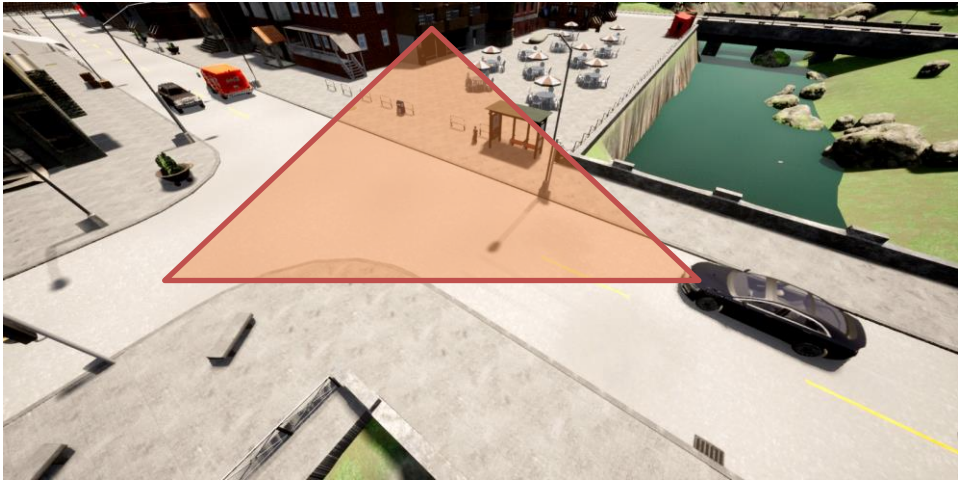
Difficult for one entity to capture all modes (***unsafe***)
Bulk of data is kept private (***inaccessible, unsafe***)
Redundancy and high cost (***slow progress, urgent application***)

Q: How to ***efficiently*** learn a navigation policy?

A: ***Learn by watching (third-person) agents*** to leverage all available demonstration sources in a scene.

Zhang and Ohn-Bar, Learning by Watching, CVPR 2021

Learning from All Humans in the Scene



Zhang and Ohn-Bar, 2021

Learning to Drive by Watching

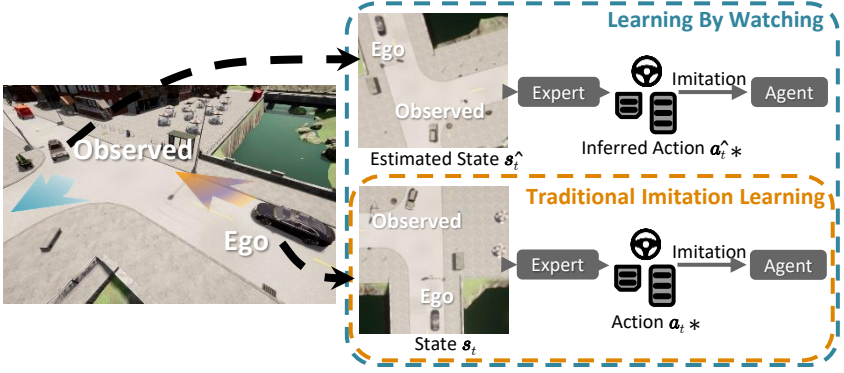
Observations: $\boldsymbol{s} = [\mathbf{I}, v]$ (image, speed)

Command: $c \in \mathcal{C} = \{$ Left, Right, Forward $\}$

Actions: $\boldsymbol{a} \in \mathcal{A} = [-1, 1]^2$

Policy: $\pi(\boldsymbol{a}|\boldsymbol{s})$

Trajectories: $\boldsymbol{\tau}^{observed} = \{(\hat{\boldsymbol{s}}_t, \hat{\boldsymbol{a}}_t^*)\}_{t=1}^{T}$



Trajectories: $\boldsymbol{\tau}^{expert} = \{(\boldsymbol{s}_t, \boldsymbol{a}_t^*)\}_{t=1}^{T}$

Imitation in Humans vs. Imitation in Robotics Today

No direct knowledge of expert actions or state

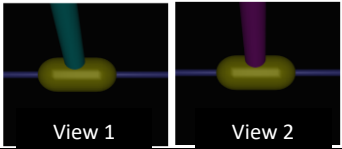
Jones SS. The development of imitation in infancy. 2009
Bandura A. Social Learning Theory.

Imitation in Humans vs. Imitation in Robotics Today

No direct knowledge of expert actions or state

Jones SS. The development of imitation in infancy. 2009
Bandura A. Social Learning Theory.

Full access to sensory state, ego-expert actions

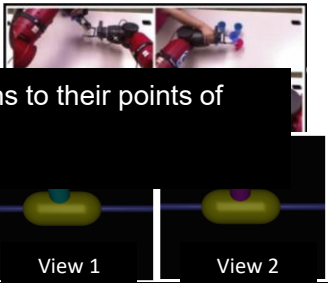


Third-Person Imitation Learning
Stadie et al., ICLR 2017
Sharma et al., NeurIPS 2019

Imitation in Humans vs. Imitation in Robotics Today

No direct knowledge of expert actions or sta...

Full access to sensory state, ego-expert actions

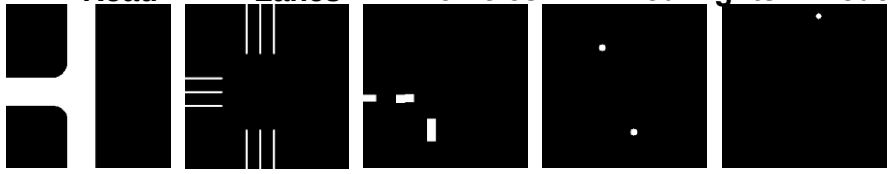


(1) Humans transform observations to their points of view
(2) Infer others' expert actions

Jones SS. The development of imitation in infancy. 2009
Bandura A. Social Learning Theory.

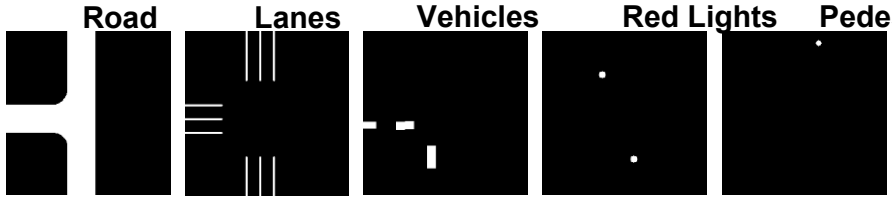
Third-Person Imitation Learning
Stadie et al., ICLR 2017
Sharma et al., NeurIPS 2019

Compact Bird's-Eye-View (BEV) State Representation

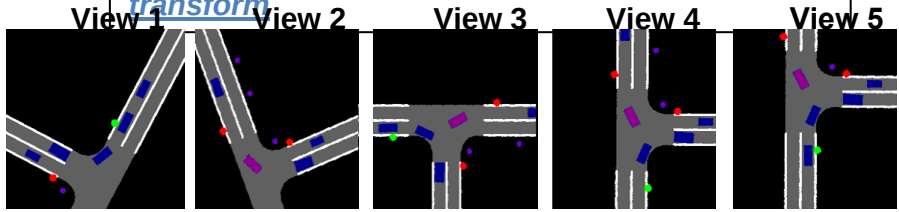


Enables relatively straightforward ***perspective transform***

Compact Bird's-Eye-View (BEV) State Representation



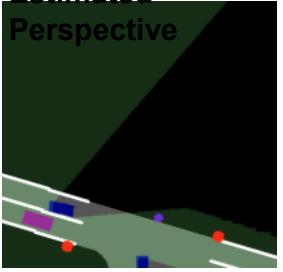
Enables relatively straightforward ***perspective transform***



Missing information regrading the ***original***

Handling Occluded Regions With Visibility Map

Percepti
on



Waypoint-Based Action Representation

$$\boldsymbol{\tau}^{observed} = \{(\hat{\mathbf{s}}_t, \hat{\mathbf{a}}_t^*)\}_{t=1}^{T}$$

Waypoint-Based Action Representation

$$\tau^{observed} = \{(\hat{\mathbf{s}}_t, \hat{\mathbf{a}}_t^*)\}_{t=1}^{T} \blacktriangleright \quad \tau^{observed} = \{(\hat{\mathbf{s}}_t, \hat{\mathbf{w}}_t^*)\}_{t=1}^{T}$$

Waypoint-Based Action Representation

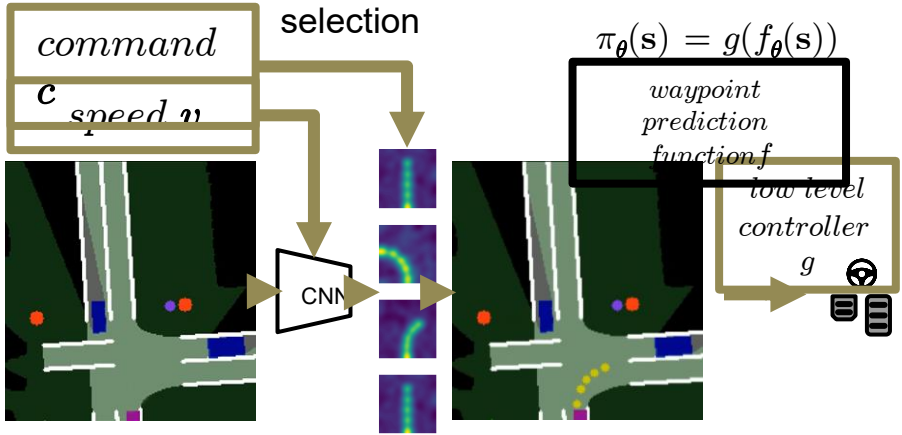
$$\tau^{observed} = \{(\hat{\mathbf{s}}_t, \hat{\mathbf{a}}^*_t)\}_{t=1}^{T} \blacktriangleright \quad \tau^{observed} = \{(\hat{\mathbf{s}}_t, \hat{\mathbf{w}}^*_t)\}_{t=1}^{T}$$

$$\pi_{\boldsymbol{\theta}}(\mathbf{s}) = g(f_{\boldsymbol{\theta}}(\mathbf{s}))$$

waypoint prediction function f

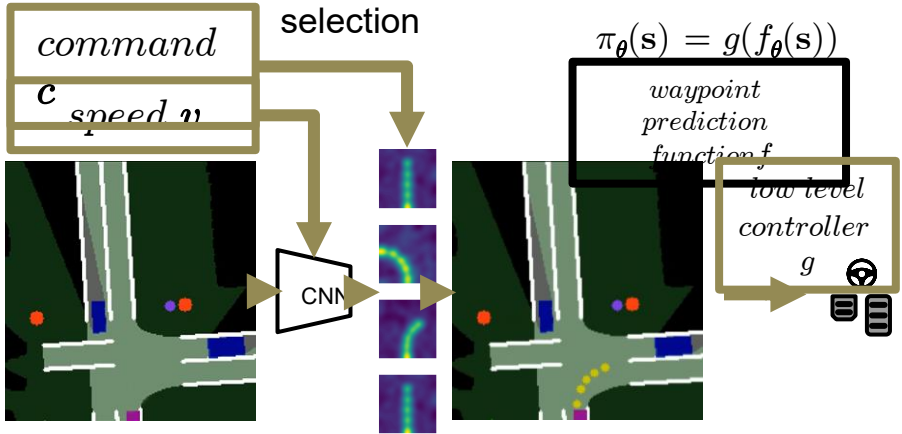
Waypoint-Based Action Representation

$$\tau^{observed} = \{(\hat{\mathbf{s}}_t, \hat{\mathbf{a}}_t^*)\}_{t=1}^{T} \quad \tau^{observed} = \{(\hat{\mathbf{s}}_t, \hat{\mathbf{w}}_t^*)\}_{t=1}^{T}$$



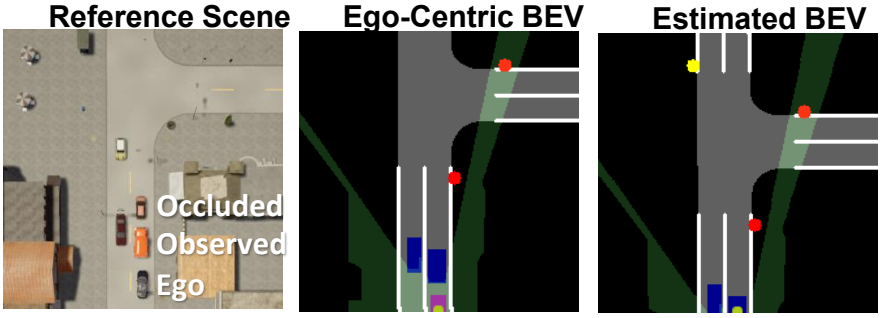
Waypoint-Based Action Representation

$$\tau^{observed} = \{(\hat{\mathbf{s}}_t, \hat{\mathbf{a}}_t^*)\}_{t=1}^{T} \blacktriangleright \quad \tau^{observed} = \{(\hat{\mathbf{s}}_t, \hat{\mathbf{w}}_t^*)\}_{t=1}^{T}$$



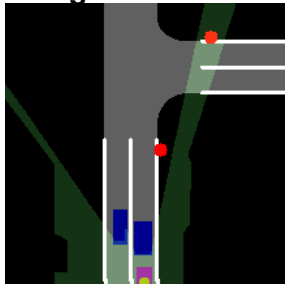
$$\mathcal{L}_{\text{behavior-cloning}} = \mathbb{E}_{(\mathbf{s},\mathbf{w})\sim\mathbf{D}}[\ell_1(\mathbf{w},\ f_{\boldsymbol{\theta}}(\mathbf{s}))]$$

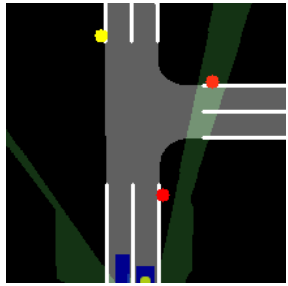
Aligning and De-Noising Samples by Observed Vehicles



ed BEV

Aligning and De-Noising Samples by Observed Vehicles

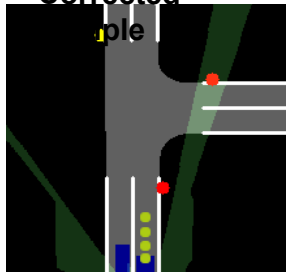




1. *Train baseline model*
2. *Predict and correct observed waypoints*

$$\hat{\mathbf{w}}^* = \beta\hat{\mathbf{w}}^* + (1-\beta)\hat{\mathbf{w}}_{ego}$$

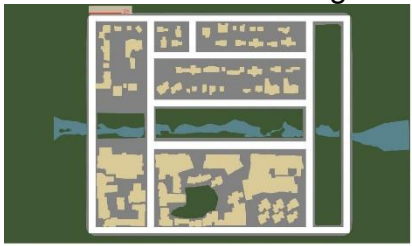
3. *Retrain with* ***entire dataset***



ed BEV

Low-Data Regime on CARLA Benchmark

Town 1: Training

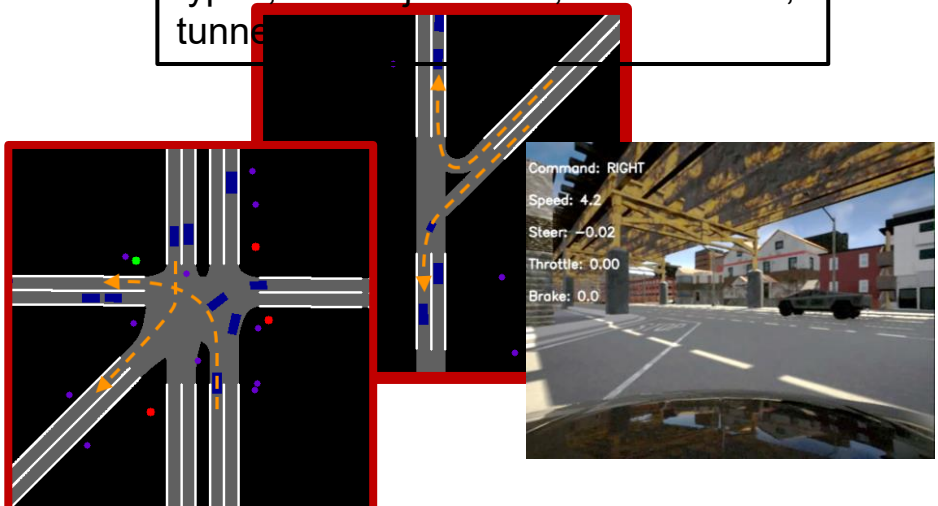


Town 2: Testing

Metric: Percentage of successfully completed episodes (**success rate**)

Low-Data Adaptation Benchmark

Town 3: Novel routes, intersection types, 5-lane junctions, roundabouts, tunne



Results

Model	One Hour Training	10 Minutes Training	Town 3 Testing
Ego (Baseline)	46	24	40
LbW	64	34	60
LbW + Visibility (Early)	52	28	60
LbW + Visibility (Late)	**92**	**52**	**100**

Baseline

LbW

Results

Model	One Hour Training	10 Minutes Training	New Town
Ego (Baseline)	46	24	40
LbW	64	34	60
LbW + Visibility (Early)	52	28	60
LbW + Visibility (Late)	**92**	**52**	**100**

Baseline

LbW


Driving Results on Town 2
(*10 Minutes* of Policy Training Data)

Conditional Imitation Learning

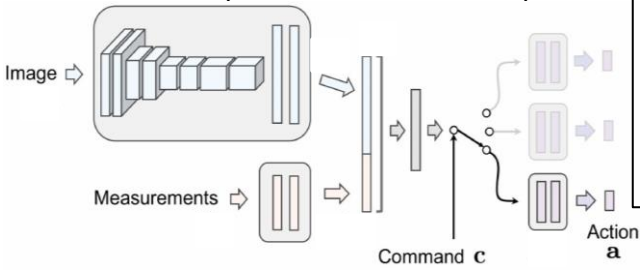
Observations: $s = [\mathbf{I}, v]$ (image, speed)

Command: $c \in \mathcal{C} = \{$ Left, Right, Forward $\}$

Actions: $a \in \mathcal{A} = [-1, 1]^2$

Policy: $\pi(a|s, c)$

Architecture (trained end-to-end):



Optimization?
Generalization?
Task supervision?
Safety?
Covariate shift?
Catastrophic failure?

Codevllia, et al., ICCV'19,

Conditional Imitation Learning

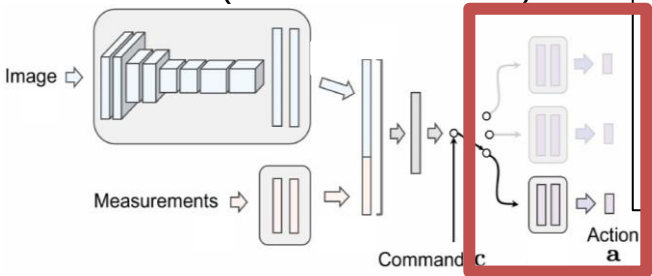
Observations: $s = [\mathbf{I}, v]$ (image, speed)

Command: $c \in \mathcal{C} = \{$ Left, Right, Forward $\}$

Actions: $a \in \mathcal{A} = [-1, 1]^2$

Policy: $\pi(a|s, c)$

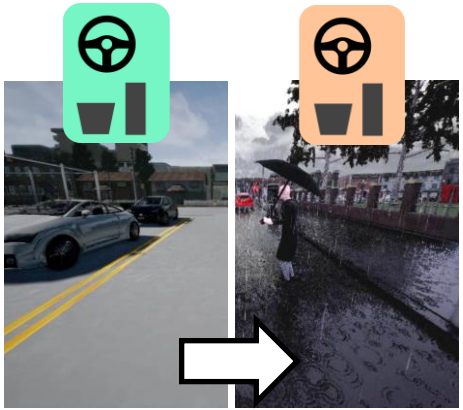
Architecture (trained end-to-end):



Optimization?
Generalization?
Task supervision?
Safety?
Covariate shift?
Catastrophic failure?

Codevllia, et al., ICCV'19,

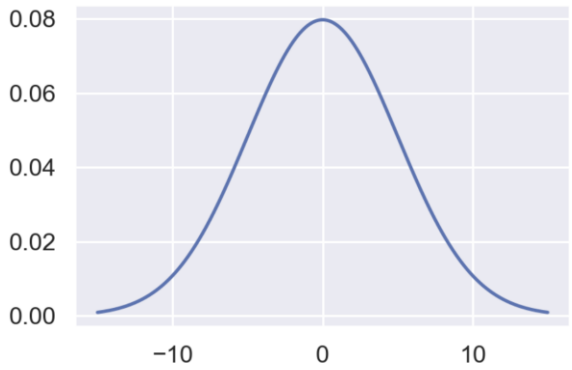
Humans Use Situation-Specific Awareness and Strategies



M. R. Endsley. Toward a Theory of Situation Awareness in Dynamic Systems, 2000.
C. C. Macadam. Understanding and modeling the human driver. Vehicle System Dynamics, 2003.

Most networks fit a Gaussian to labels

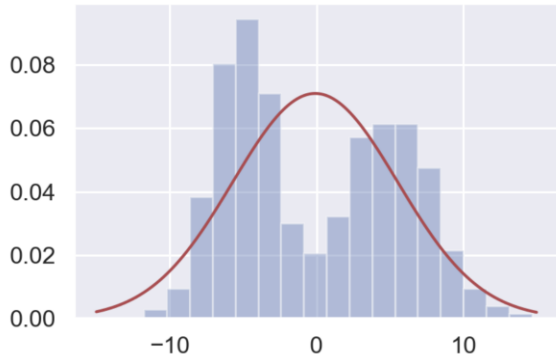
- “Standard” probability distribution
- Has two parameters:
  - mean (μ) and
  - standard deviation (σ)
- Probability Density Function:
  - $$N(x \mid \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$



PROBLEM: NORMAL DISTRIBUTION MIGHT NOT FIT DATA

What if the data is complicated?

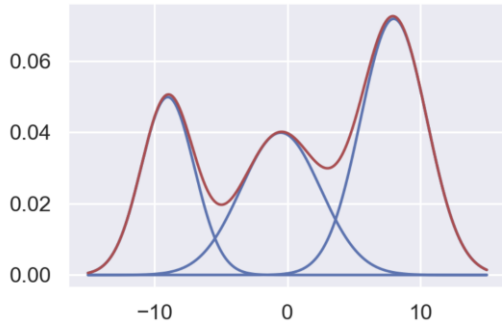
- It's easy to "fit" a normal model to any data.
  - Just calculate μ and σ
- But this might not fit the data well.



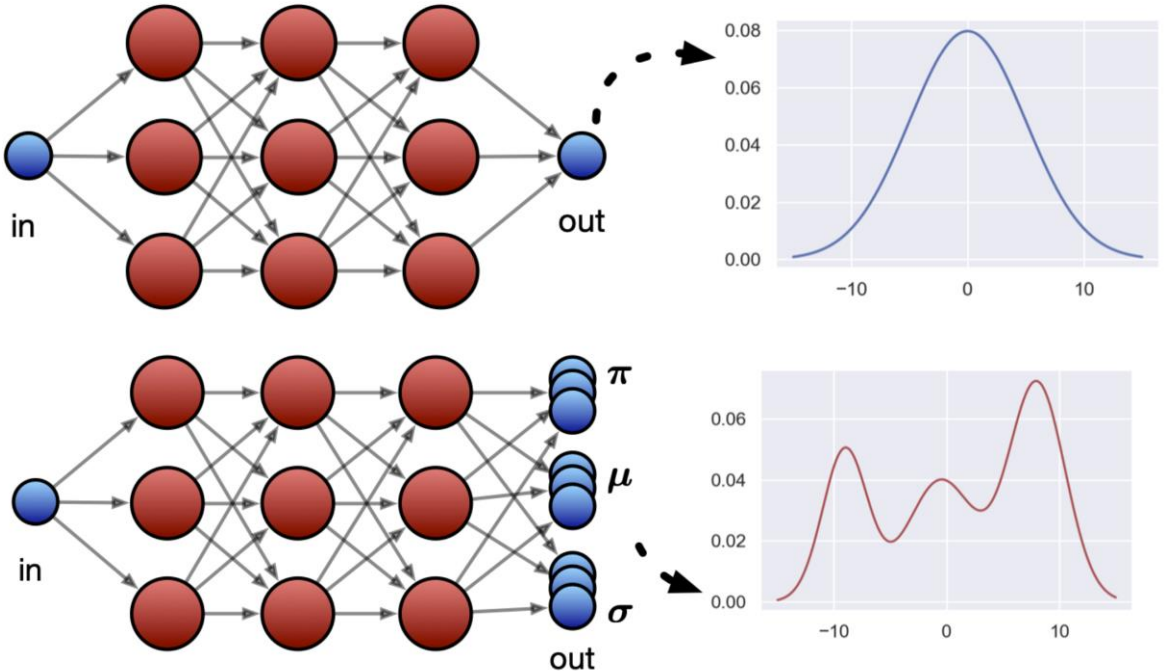
MIXTURE OF NORMALS

Three groups of parameters:

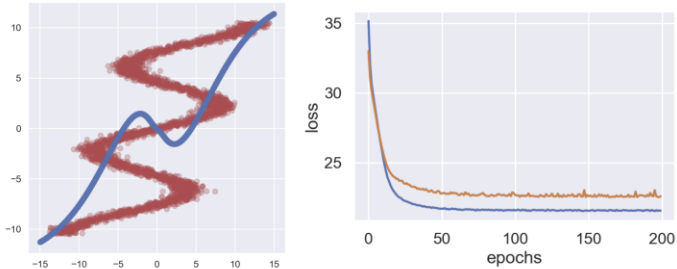
- means (μ): location of each component
- standard deviations (σ): width of each component
- Weight (π): height of each curve
- Probability Density Function:
  - $$p(x) = \sum_{i=1}^{K} \pi_i N(x \mid \mu, \sigma^2)$$



Mixture Density Networks, Bishop, 1994


in
out
0.08
0.06
0.04
0.02
0.00
−10
0
10
in
π
μ
σ
out
0.06
0.04
0.02
0.00
−10
0
10

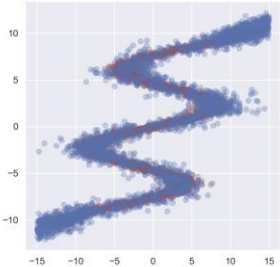
FEEDFORWARD MSE NETWORK

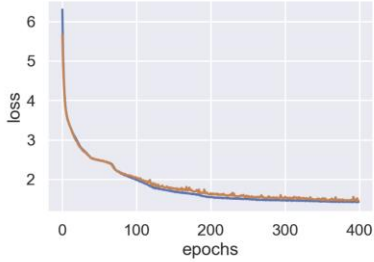


Here's a simple two-hidden-layer network (286 parameters), trained to produce the above result.

```
model = Sequential()
model.add(Dense(15, batch_input_shape=(None, 1), activation='tanh'))
model.add(Dense(15, activation='tanh'))
model.add(Dense(1, activation='linear'))
model.compile(loss='mse', optimizer='rmsprop')
model.fit(x=x_data, y=y_data, batch_size=128, epochs=200, validation_split=0.15)
```

FEEDFORWARD MDN SOLUTION



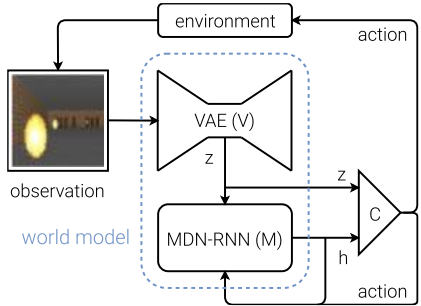


And, here's a simple two-hidden-layer MDN (510 parameters), that achieves the above result! Much better!

```
N_MIXES = 5

model = Sequential()
model.add(Dense(15, batch_input_shape=(None, 1), activation='relu'))
model.add(Dense(15, activation='relu'))
model.add(mdn.MDN(1, N_MIXES)) # here's the MDN layer!
model.compile(loss=mdn.get_mixture_loss_func(1,N_MIXES), optimizer='rmsprop')
model.summary()
```

Inspiration: World Models (Didn't Work Very Well...)



Step 1: Learn **generative model** of game environment (VAE)

Step 2: Learn dynamics and control models in **latent space** (CMA-ES)

Method	Score
DQN	343
A3C	652
World Models	**906**

Fails beyond simplest settings. ***Mixture Density Network (MDN) is key***.

Ha and Schmidhuber, Recurrent World Models Facilitate Policy Evolution , NeurIPS 2018

Proposed ***Learning Situational Driving (LSD)*** Framework

Observations: $\boldsymbol{s} = [\mathbf{I}, v]$ (image, speed)

Command: $c \in \mathcal{C} = \{$ Left, Right, Forward $\}$

Actions: $\boldsymbol{a} \in \mathcal{A} = [-1, 1]^2$

Policies: $\Pi = \{{}^1\pi_\theta, \ldots, {}^K\pi_\theta\}$

Parameters: $\Theta = \{\theta, \phi, \boldsymbol{\Psi}\}$

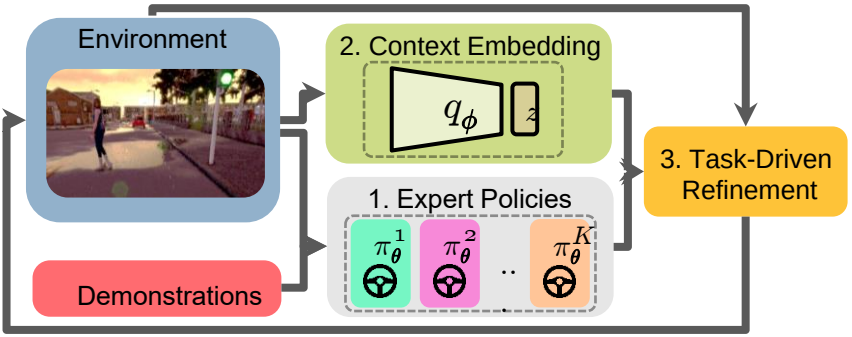
$$\pi_\Theta(\boldsymbol{a}|\boldsymbol{s}, c) = \sum_{k=1}^{K} \underbrace{\alpha_\theta^k(\boldsymbol{s}, c)}_{\text{Mixture Weights}} \underbrace{\pi_\theta^k(\boldsymbol{a}|\boldsymbol{s}, c)}_{\textbf{\textit{Expert Models}}} + \boldsymbol{\Psi} \underbrace{\begin{bmatrix} q_\phi(\mathbf{I}) \\ v \\ c \end{bmatrix}}_{\text{Context Embedding}}$$

$$\pi_\theta^k(\boldsymbol{a}|\boldsymbol{s}, c) = \mathcal{N}\left(\boldsymbol{a}|\boldsymbol{\mu}_\theta^k(\boldsymbol{s}, c), \text{diag}(\boldsymbol{\sigma}_\theta^k(\boldsymbol{s}, c)^2)\right)$$

Key Idea

Learn **specialized experts** and combine in a context-dependent

Multistep Model Training and Optimization



Step 1: Learn **mixture of experts** via imitation $\mathcal{L}_{\text{MoE}} = -\log[\sum_{k=1}^{K} \alpha_\theta^k \pi_\theta^k]$

Step 2: Learn a general purpose context encoder q_ϕ as a β-VAE

$$\mathcal{L}_{\text{VAE}} = \beta\ \text{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\mathbf{I})\ ||\ \mathcal{N}_0)\ +\ ||d_{\boldsymbol{\phi}}(\boldsymbol{z})\ -\ \mathbf{I}||^2$$

Step 3: **Task-driven** policy refinement with CMA-ES for $\mathbb{E}_{\pi_\Theta}[\sum_t r_t]$

CARLA Benchmark

Metric: Percentage of successfully completed episodes (**success rate**)

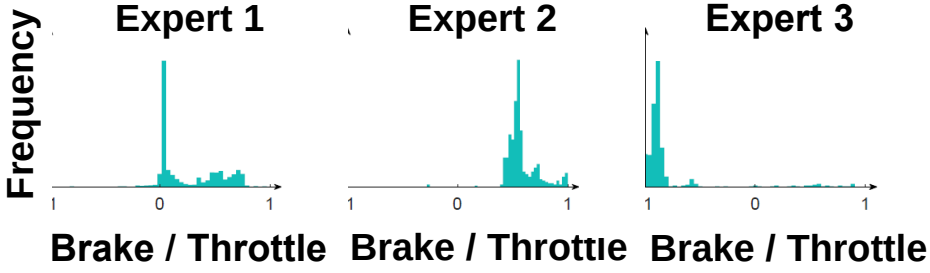
Town 1: Training

Town 2: Testing

Results & Emergent Driving Behavior Modes

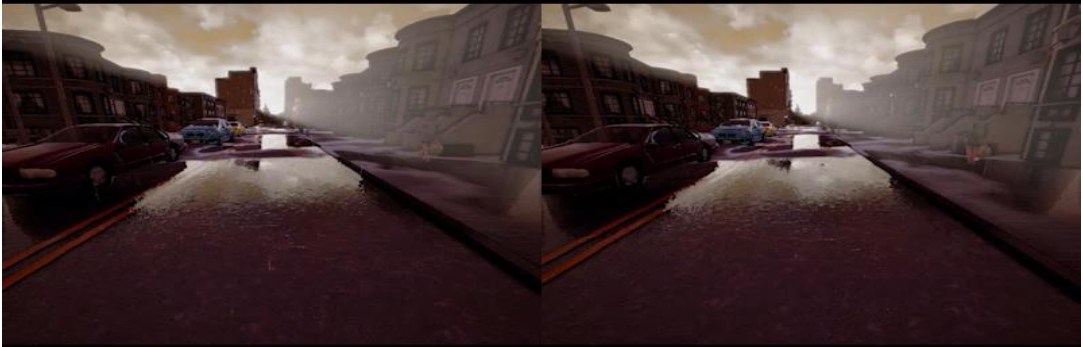
Driving Task	CIRL (ECCV18)	CILRS (ICCV19)	LSD (ours, step 1)	LSD+Reward (ours, steps 1-3)
Straight (Static)	**100**	96	**100**	**100**
One Turn (Static)	71	84	**100**	**100**
Navigation (Static)	53	69	**98**	**100**
Dense Traffic	41	66	92	**98**

Acceleration distribution of three different experts during testing



LSD
(***Times-out*** due
to reflection)

LSD+Reward
(***Success***)



Further Readings

- Learning by Cheating, Conference on Robot Learning, 2019
- Learning by Watching, CVPR, 2021
- Learning monocular reactive UAV control in cluttered natural environments, ICRA 2013
- Learning Situational Driving, CVPR, 2020