Carnegie Mellon
School of Computer Science

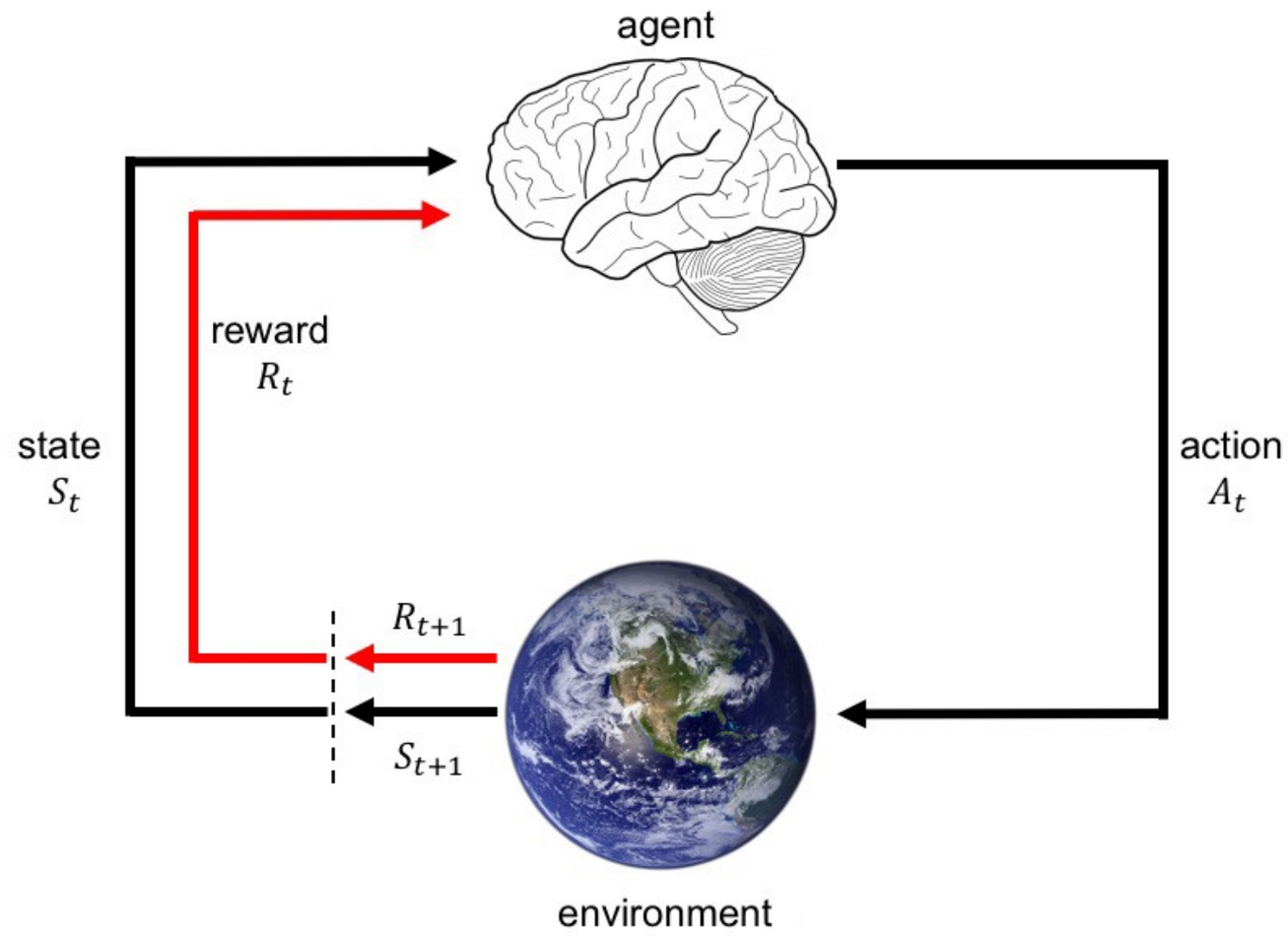
Deep Reinforcement Learning and Control

Imitation Learning

Fall 2019, CMU 10-703

Katerina Fragkiadaki

Reinforcement learning



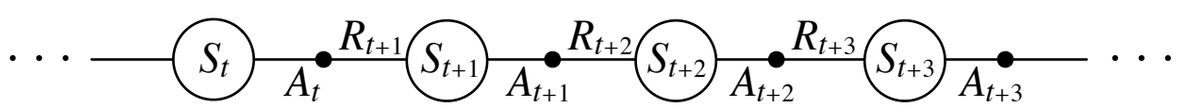
Agent and environment interact at discrete time steps: $t = 0, 1, 2, 3, \ldots$

Agent observes state at step t: $S_t \in \mathcal{S}$

produces action at step t : $A_t \in \mathcal{A}(S_t)$

gets resulting reward: $R_{t+1} \in \mathcal{R} \subset \mathbb{R}$

and resulting next state: $S_{t+1} \in \mathcal{S}^+$



Limitations of Learning by Interaction

- The agent should have the chance to try (and fail) MANY times
- This is impossible when safety is a concern: we cannot afford to fail
- This is also quite impossible in general in real life where each interaction takes time (in contrast to simulation)

Crusher robot

Learning from Demonstration for Autonomous Navigation in Complex Unstructured Terrain, Silver et al. 2010

Imitation Learning (a.k.a. Learning from Demonstrations)

visual imitation

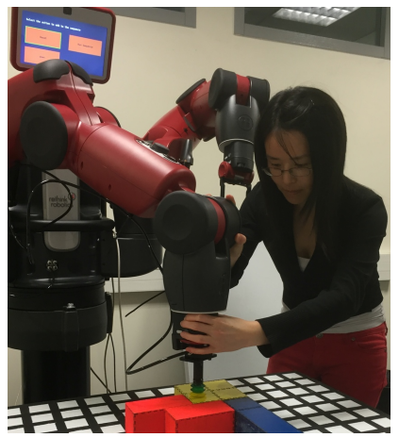
The actions of the teacher need to be inferred from visual sensory input and mapped to the end-effectors to the agent.

Two challenges:

1) visual understanding
2) action mapping, especially when the agent and the teacher do not have the same action space

we will come back to this in a later lecture

kinesthetic imitation



- The teacher takes over the end-effectors of the agent.
- Demonstrated actions can be imitated directly (*cloned*)
- A.k.a. behavior cloning

this lecture!

Imitating Controllers

visual imitation

kinesthetic imitation

- Experts do not need to be humans.
- Machinery that we develop in this lecture can be used for imitating expert policies found through (easier) optimization in a constrained smaller part of the state space.
- Imitation then means distilling knowledge of expert constrained policies into a general policy that can do well in all scenarios the simpler policies do well.

The actions of the teacher need to be inferred from visual sensory input and mapped to the end-effectors to the agent.

Two challenges:

1) visual understanding
2) action mapping, especially when the agent and the teacher do not have the same action space

We will come back to this in a later lecture

- The teacher takes over the end-effectors of the agent.
- Demonstrated actions can be imitated directly (cloned)
- A.k.a. behavior cloning

this lecture!

Notation

Richard Bellman

Lev Pontryagin

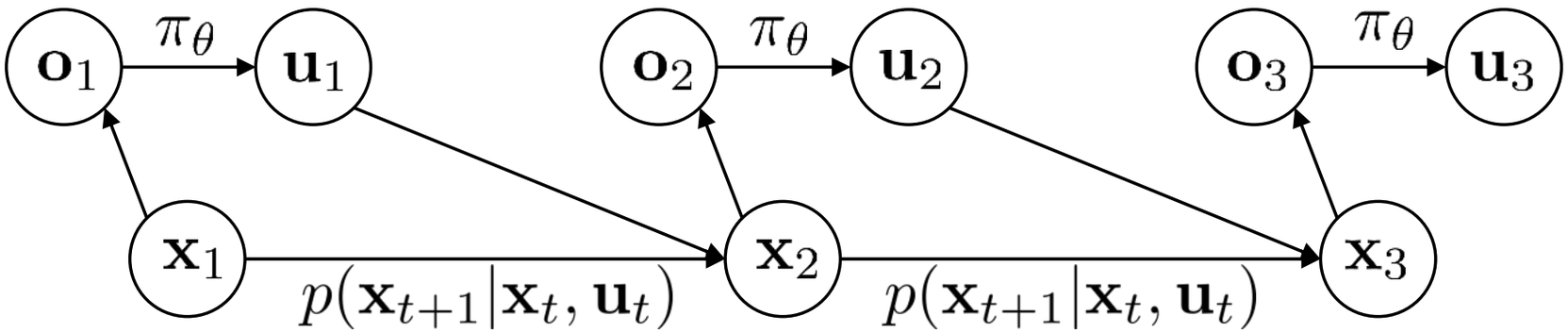
actions	a_t
states	s_t
rewards	r_t
dynamics	$p(s_{t+1} \mid s_t, a_t)$
observations	o_t

actions	u_t
states	x_t
costs	$c(x_t, u_t)$
dynamics	$p(x_{t+1} \mid x_t, u_t)$

Diagram from Sergey Levine

Imitation learning VS Sequence labelling

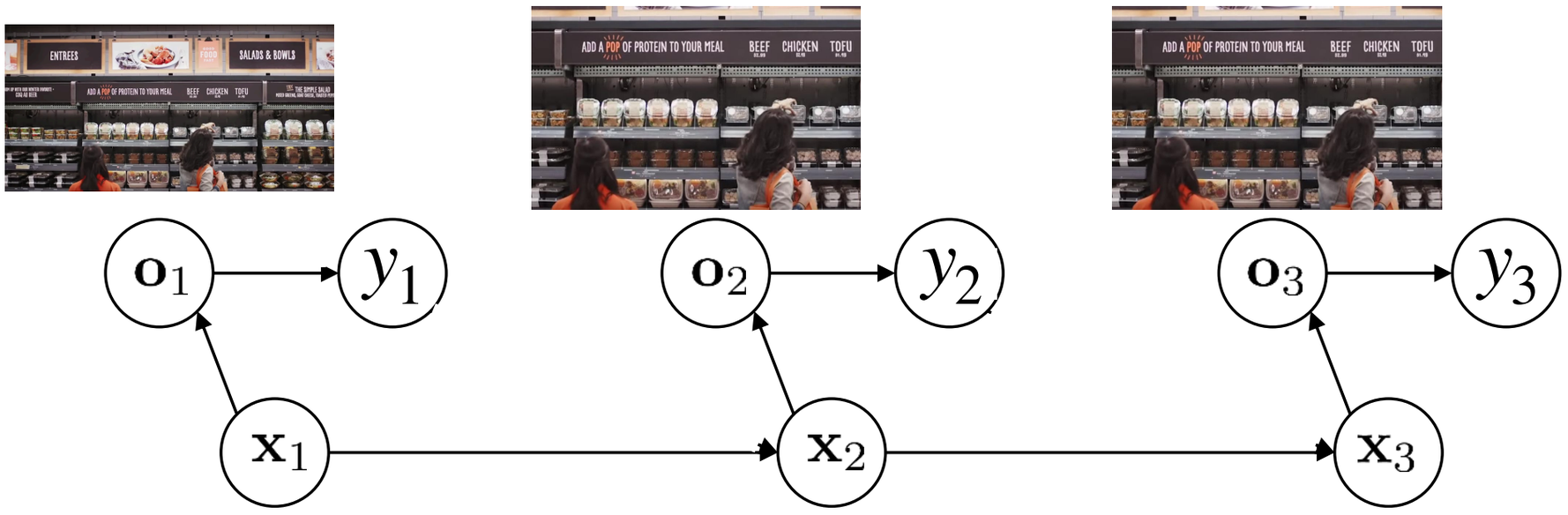
Imitation learning



Training data:

$o_1^1, u_1^1, o_2^1, u_2^1, o_3^1, u_3^1, \ldots.$
$o_1^2, u_1^2, o_2^2, u_2^2, o_3^2, u_3^2, \ldots.$
$o_1^3, u_1^3, o_2^3, u_2^3, o_3^3, u_3^3, \ldots.$

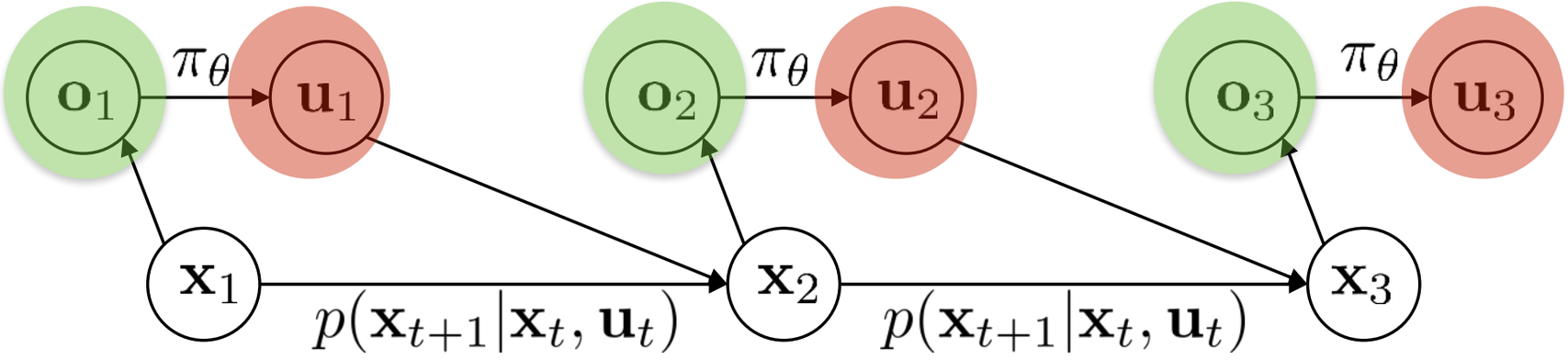
Sequence labelling



y: which product was purchased if any

Imitation learning VS Sequence labelling

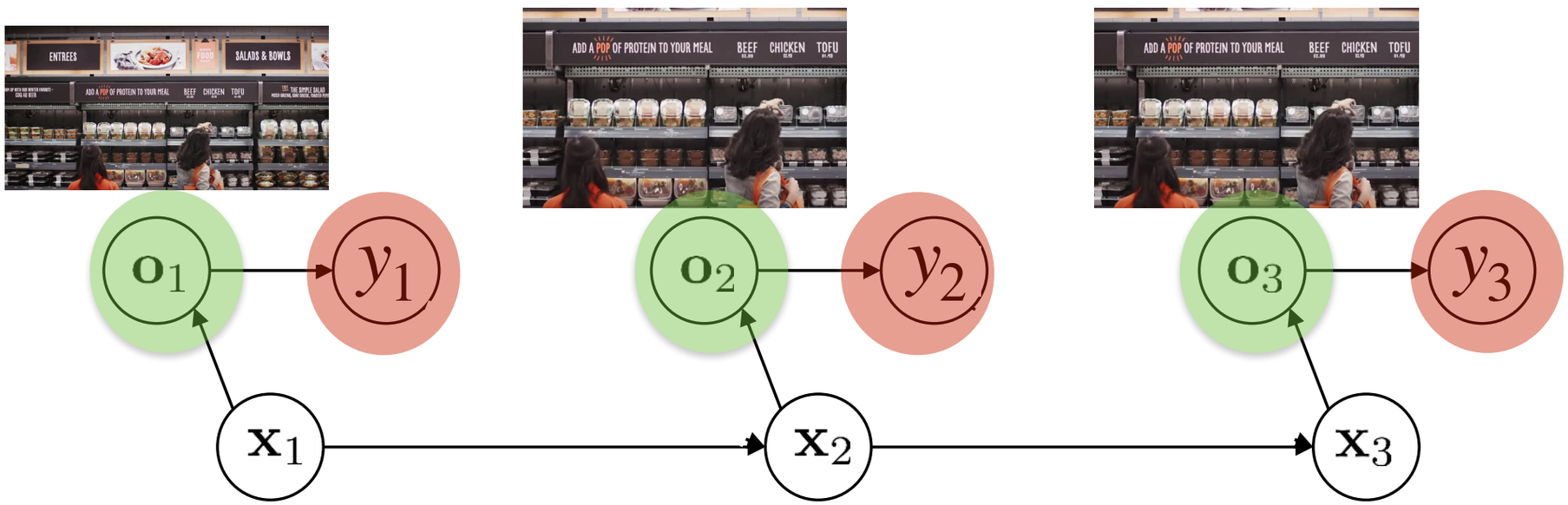
Imitation learning



Training data:

$o_1^1, u_1^1, o_2^1, u_2^1, o_3^1, u_3^1, \ldots.$
$o_1^2, u_1^2, o_2^2, u_2^2, o_3^2, u_3^2, \ldots.$
$o_1^3, u_1^3, o_2^3, u_2^3, o_3^3, u_3^3, \ldots.$

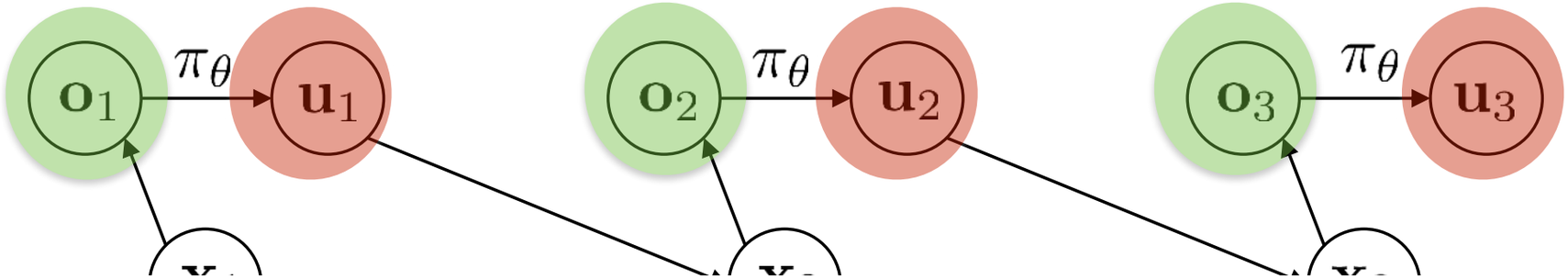
Sequence labelling



y: which product was purchased if any

Imitation learning VS Sequence labelling

Imitation learning



Action interdependence in imitation learning: the actions we predict will influence the data we will see next, and thus, our future predictions.

g data:

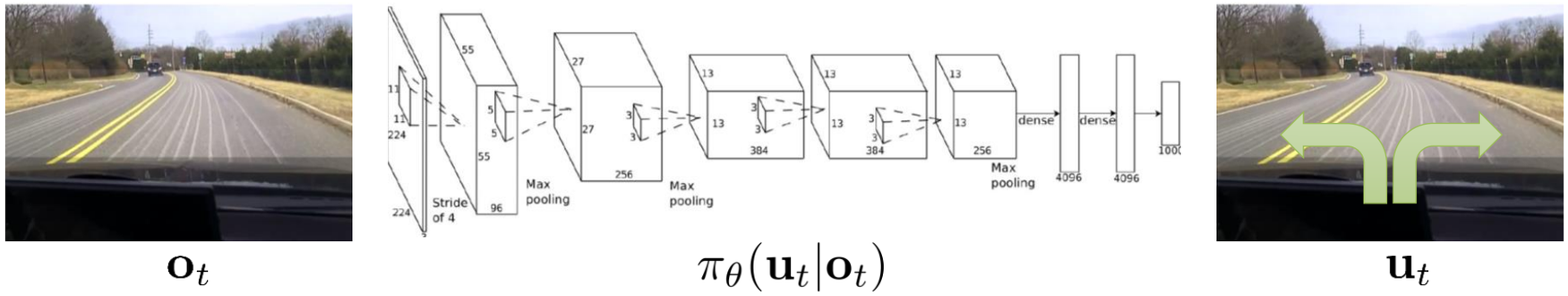
$_3^1, u_3^1, \ldots.$
$_3^2, u_3^2, \ldots.$
$_3^3, u_3^3, \ldots.$

Label interdependence is present in any structured prediction task, e.g, text generation: words we predict influence words we need to predict further down the sentence…

y: which product was purchased if any

Imitation Learning for Driving

Driving policy: a mapping from observations to steering wheel angles



$\mathbf{o}_t$ $\pi_\theta(\mathbf{u}_t|\mathbf{o}_t)$ $\mathbf{u}_t$

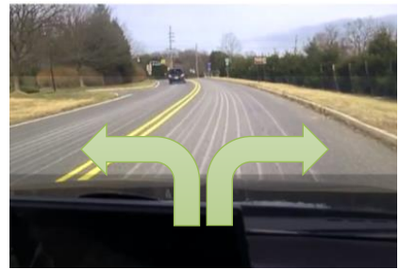
End to End Learning for Self-Driving Cars, Bojarski et al. 2016

Imitation Learning as Supervised Learning

Driving policy: a mapping from observations to steering wheel angles

$\mathbf{o}_t$

$\pi_\theta(\mathbf{u}_t|\mathbf{o}_t)$

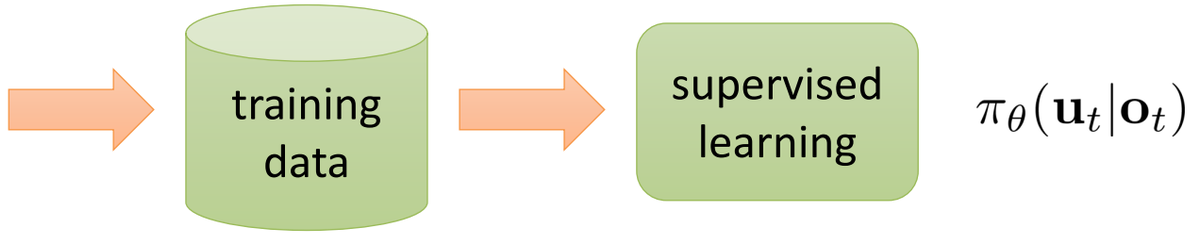


$\mathbf{u}_t$

- Assume actions in the expert trajectories are i.i.d. (independent and identically distrubuted)
- Train a function approximator to map observations to actions at each time step of the trajectory.

$\mathbf{o}_t$

$\mathbf{u}_t$



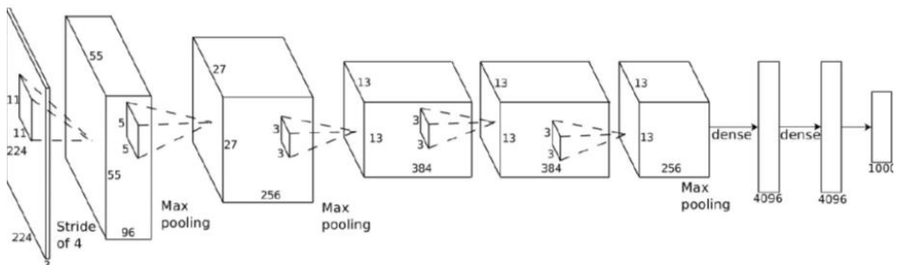
$\pi_\theta(\mathbf{u}_t|\mathbf{o}_t)$

End to End Learning for Self-Driving Cars, Bojarski et al. 2016

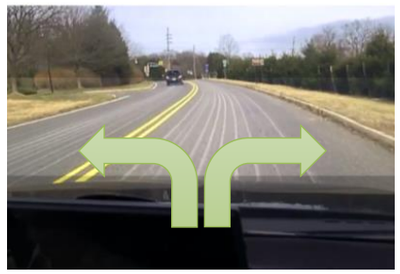
What can go wrong?

- Compounding errors
  Fix: data augmentation
- Stochastic expert actions
  Fix: stochastic latent variable models, action discretiation, gaussian mixture networks
- Non-markovian observations
  Fix: observation concatenation or recurrent models

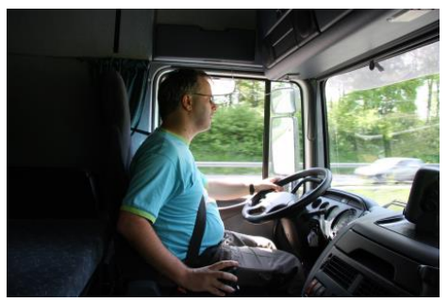
$\mathbf{o}_t$



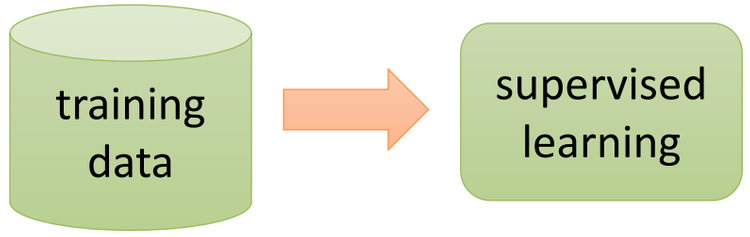
$\pi_\theta(\mathbf{u}_t|\mathbf{o}_t)$



$\mathbf{u}_t$



$\mathbf{o}_t$
$\mathbf{u}_t$



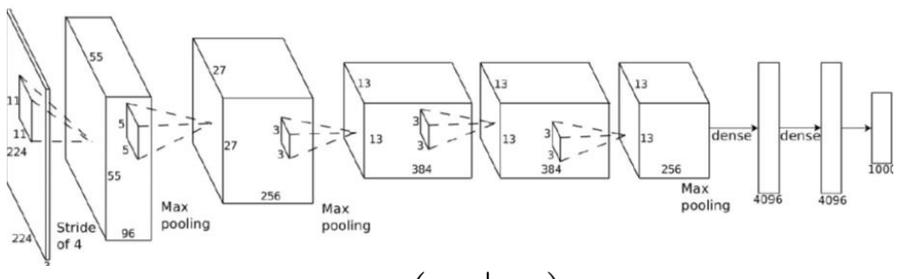
$\pi_\theta(\mathbf{u}_t|\mathbf{o}_t)$

End to End Learning for Self-Driving Cars, Bojarski et al. 2016

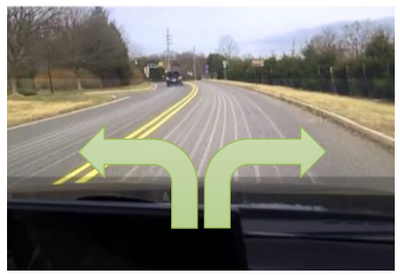
What can go wrong?

- Compounding errors
  Fix: data augmentation
- Stochastic expert actions
  Fix: stochastic latent variable models, action discretiation, gaussian mixture networks
- Non-markovian observations
  Fix: observation concatenation or recurrent models

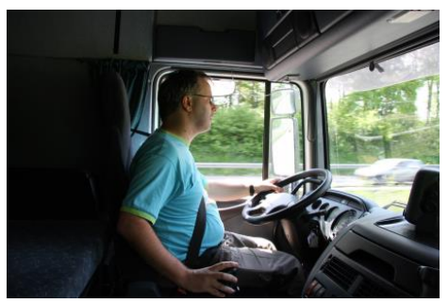
$\mathbf{o}_t$



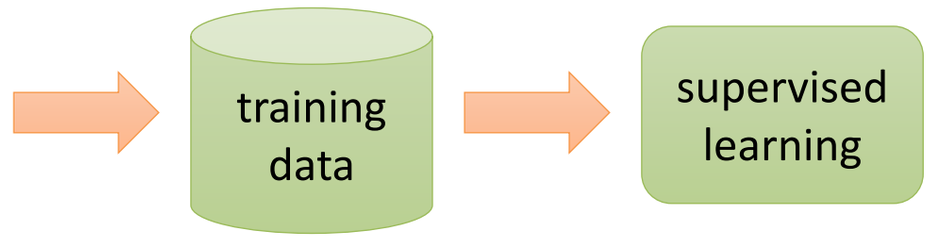
$\pi_\theta(\mathbf{u}_t|\mathbf{o}_t)$



$\mathbf{u}_t$



$\mathbf{o}_t$
$\mathbf{u}_t$

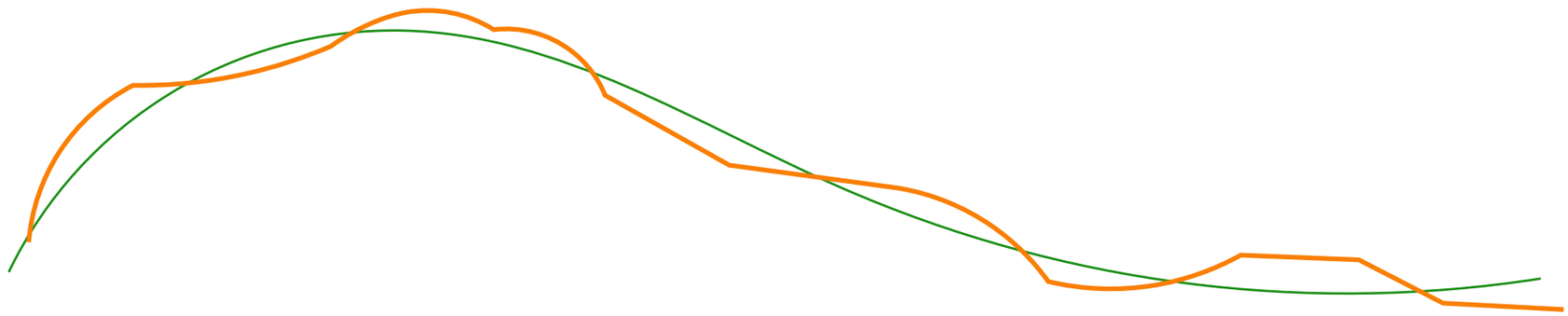


$\pi_\theta(\mathbf{u}_t|\mathbf{o}_t)$

End to End Learning for Self-Driving Cars, Bojarski et al. 2016

Independent in time errors

This means that at each time step t, the agent wakes up on a state drawn from the data distribution of the expert trajectories, and executes an action

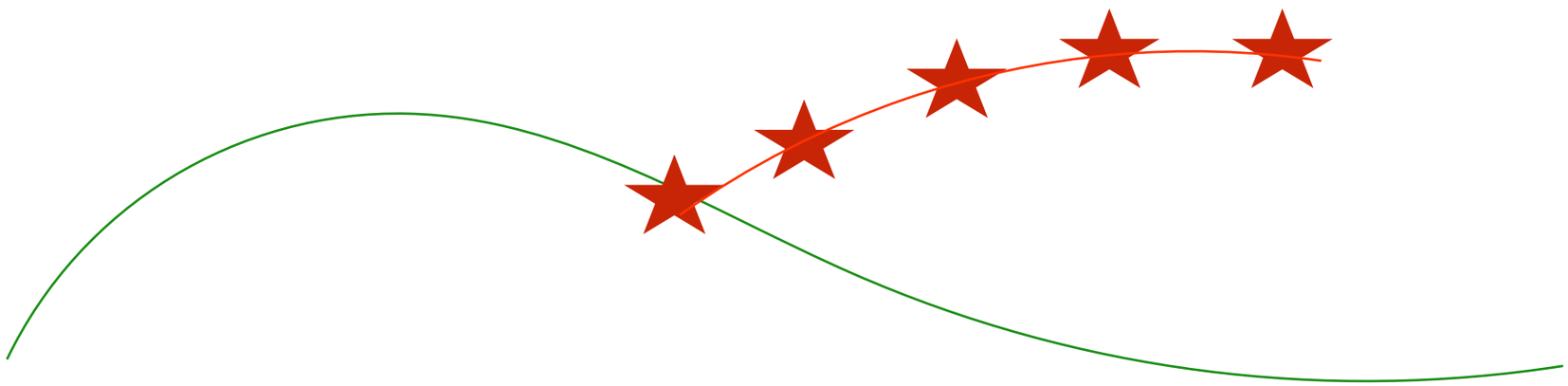


error at time t with probability ε

E[Total errors] ≲ εT

Compounding Errors

This means that at each time step t, the agent wakes up on the state that resulted from executing the action the learned policy suggested in the previous time step.



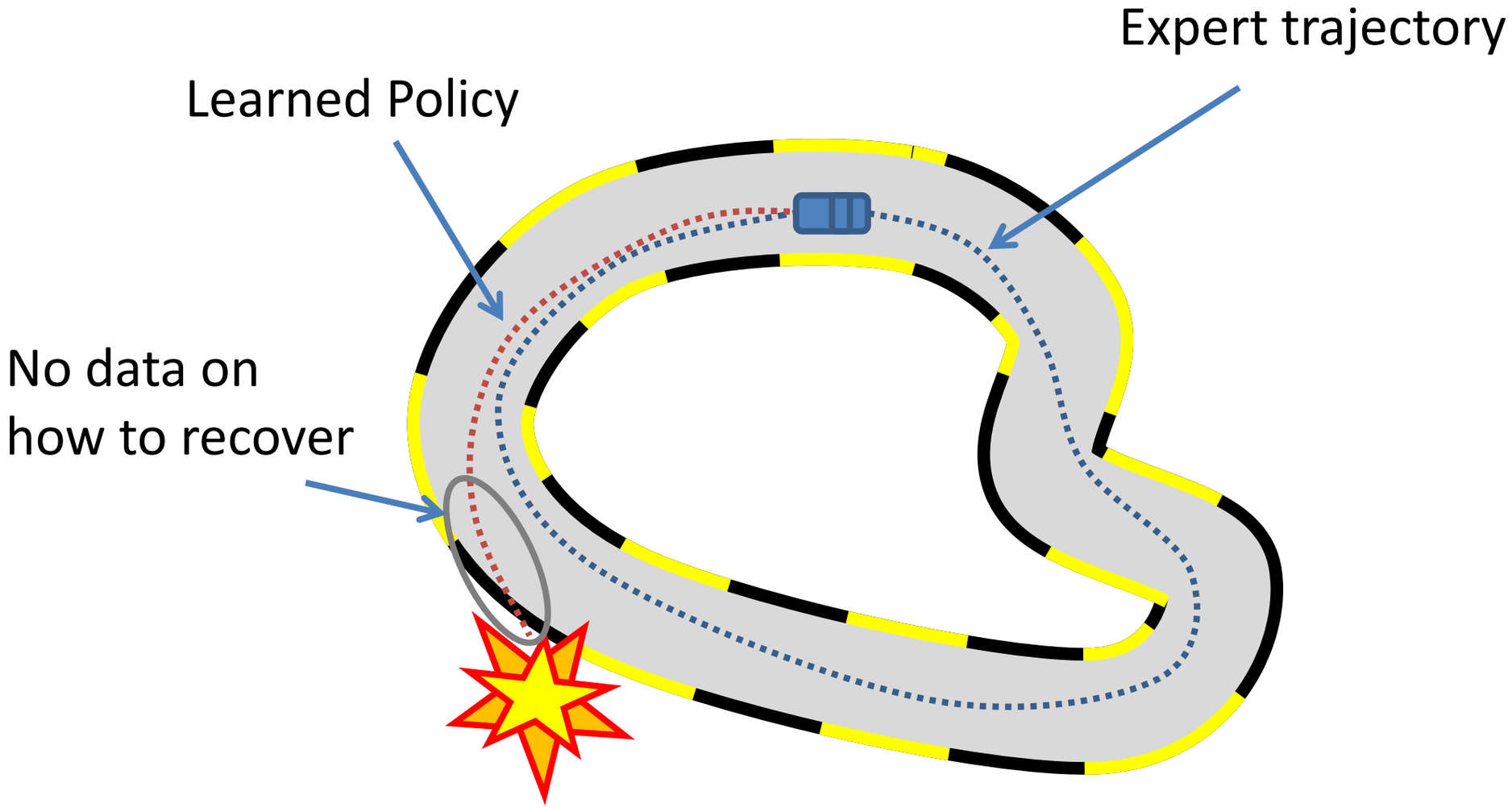
error at time t with probability ε

E[Total errors] ≤ ε(T + (T-1) + (T-2) + …+ 1) $\propto$ εT^2

A Reduction of Imitation Learning and Structured Prediction to No-Regret Online Learning, Ross et al. 2011

Data Distribution Mismatch!

$$p_{\pi^*}(o_t) \neq p_{\pi_\theta}(o_t)$$



Data Distribution Mismatch!

	supervised learning	supervised learning + control (NAIVE)
train	$(x,y) \sim D$	$s \sim d_{\pi^*}$
test	$(x,y) \sim D$	$s \sim d_{\pi}$

SL succeeds when training and test data distributions match, that is a fundamental assumption.

Solution: data augmentations

Change $p_{\pi^*}(o_t)$ using demonstration augmentation!! Add examples in expert demonstration trajectories to cover the states/observations points where the agent will land when trying out its own policy. How?

- Synthetically in simulation or by clever hardware
- Interactively with experts in the loop (DAGGER)

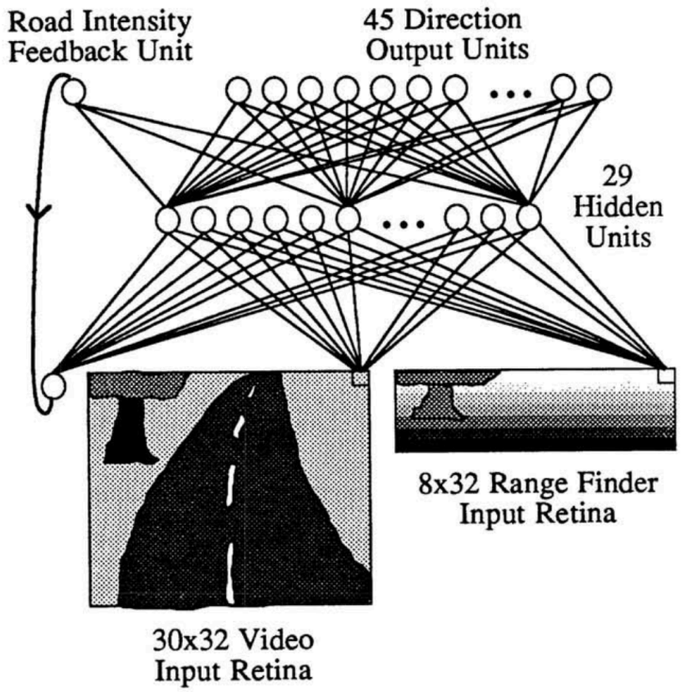
Solution: data augmentations

Change the training data distribution $p_{\pi^*}(o_t)$ using demonstration augmentation: add examples in expert demonstration trajectories to cover the states/observations where the agent will land when trying out its own policy.

	supervised learning	supervised learning + control (NAIVE)
train	(x,y) ~ D	s ~ d_{π^*}
test	(x,y) ~ D	s ~ d_{π}

Demonstration Augmentation: ALVINN 1989

Road follower

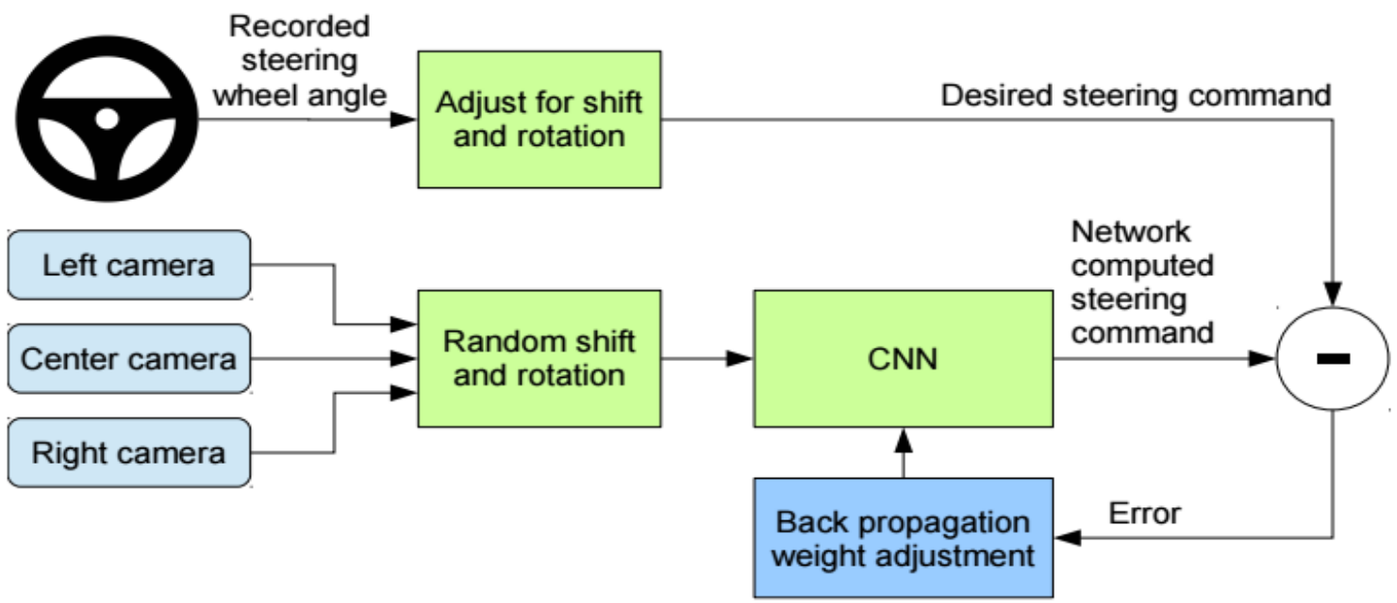


Real Road Image Simulated Road Image

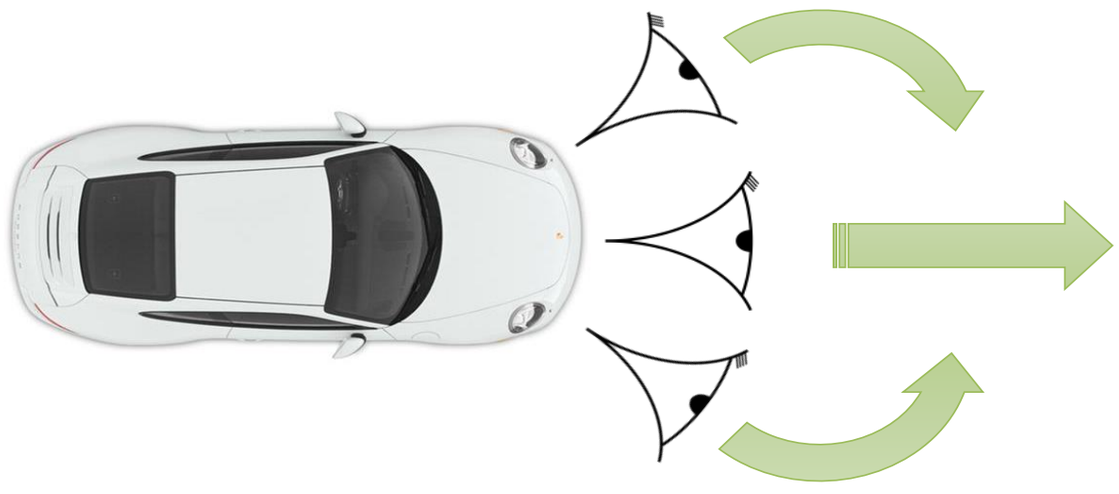
- Using graphics simulator for road images and corresponding steering angle ground-truth
- Online adaptation to human driver steering angle control
- 3 layers, fully connected layers, very low resolution input from camera

> *"In addition, the network must not solely be shown examples of accurate driving, but also how to recover (i.e. return to the road center) once a mistake has been made. Partial initial training on a variety of simulated road images should help eliminate these difficulties and facilitate better performance."*
> *ALVINN: An autonomous Land vehicle in a neural Network", Pomerleau 1989*

Demonstration Augmentation: NVIDIA 2016



Additional, left and right cameras with automatic grant-truth labels to recover from mistakes



"DAVE-2 was inspired by the pioneering work of Pomerleau [6] who in 1989 built the Autonomous Land Vehicle in a Neural Network (ALVINN) system. Training with data from only the human driver is not sufficient. The network must learn how to recover from mistakes. …"

End to End Learning for Self-Driving Cars , Bojarski et al. 2016

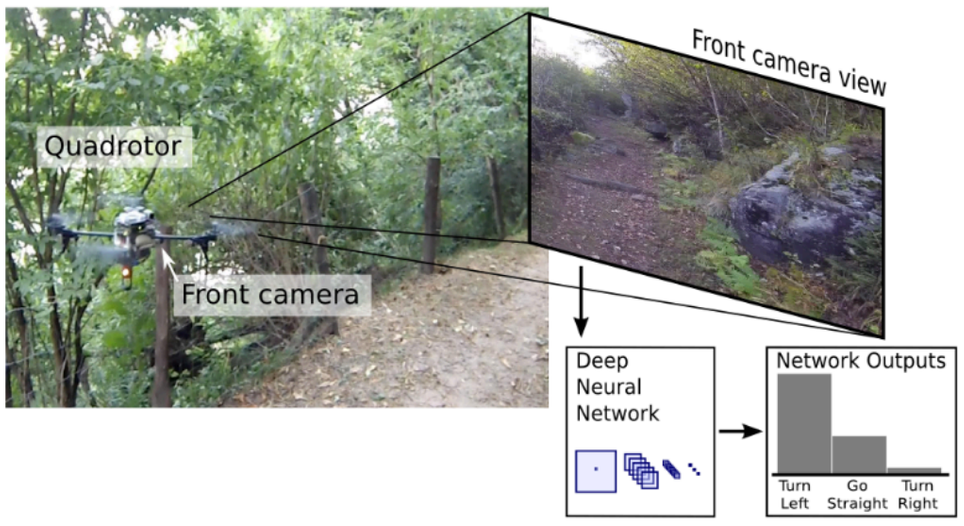
Data Augmentation (2): NVIDIA 2016

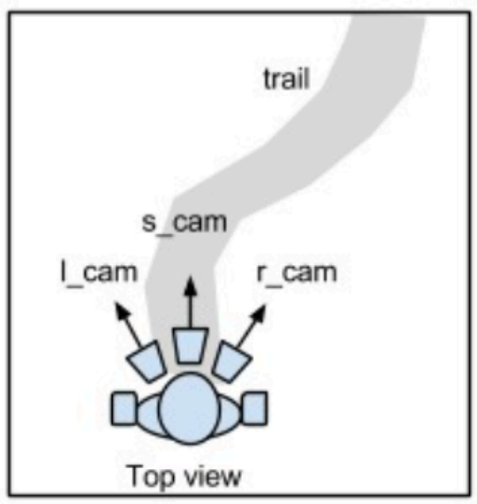
DAVE 2 Driving a Lincoln

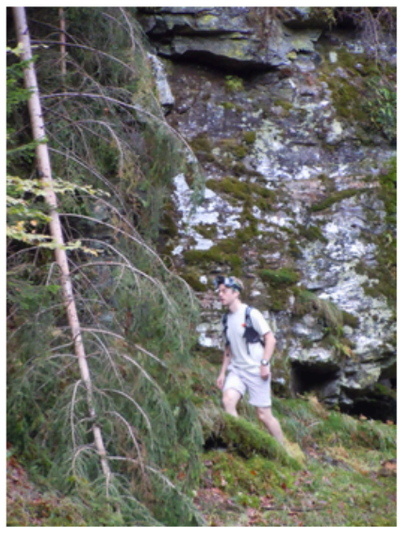
- A convolutional neural network
- Trained by human drivers
- Learns perception, path planning, and control
  "pixel in, action out"
- Front-facing camera is the only sensor

“DAVE-2 was inspired by the pioneering work of Pomerleau [6] who in 1989 built the Autonomous Land Vehicle in a Neural Network (ALVINN) system. Training with data from only the human driver is not sufficient. The network must learn how to recover from mistakes. …”, End to End Learning for Self-Driving Cars , Bojarski et al. 2016

Data Augmentation (3): Trails 2015

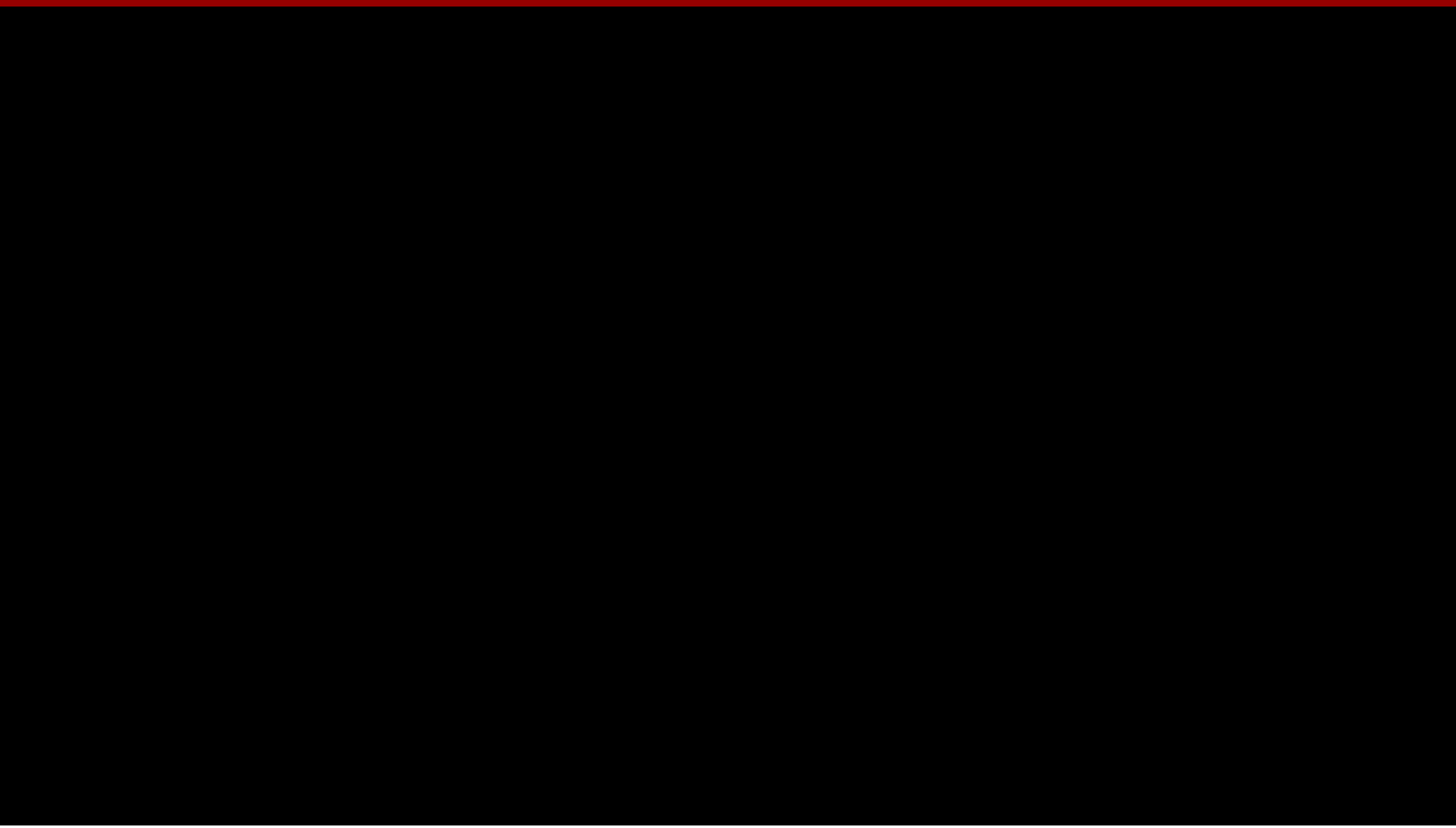






A Machine Learning Approach to Visual Perception of Forest Trails for Mobile Robots Giusti et al.

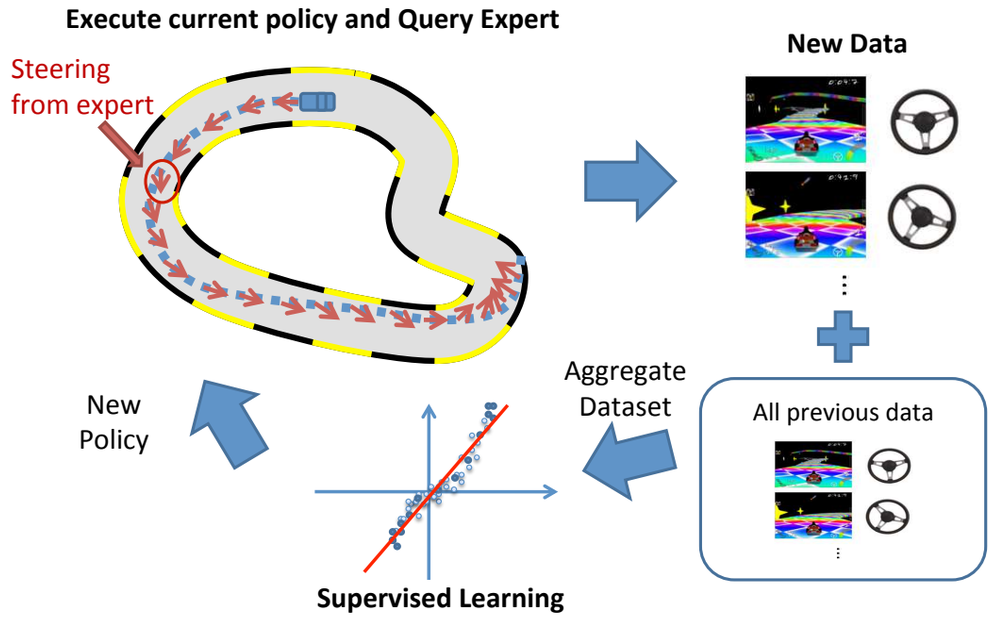
Data Augmentation (3): Trails 2015



A Machine Learning Approach to Visual Perception of Forest Trails for Mobile Robots Giusti et al.

DAGGER (in simulation)

Dataset AGGregation: bring learner's and expert's trajectory distributions closer by (asking uman experts to provide) labelling additional data points resulting from applying the current policy



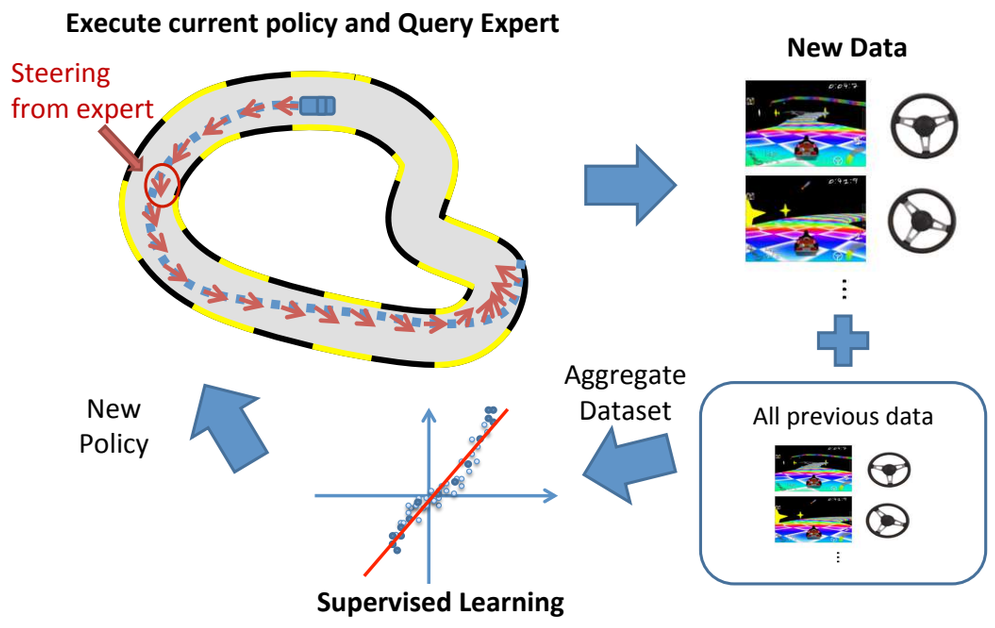
A Reduction of Imitation Learning and Structured Prediction to No-Regret Online Learning, Ross et al. 2011

DAGGER (in simulation)

Dataset AGGregation: bring learner's and expert's trajectory distributions closer by (asking uman experts to provide) labelling additional data points resulting from applying the current policy

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

$$\mathcal{D}_{\pi^*} \leftarrow \mathcal{D}_{\pi^*} \cup \mathcal{D}_\pi$$



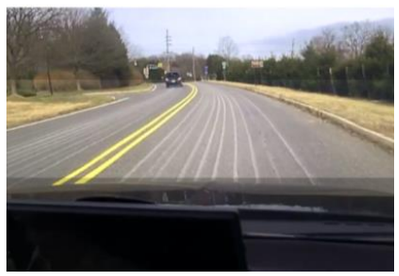
Problems:

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

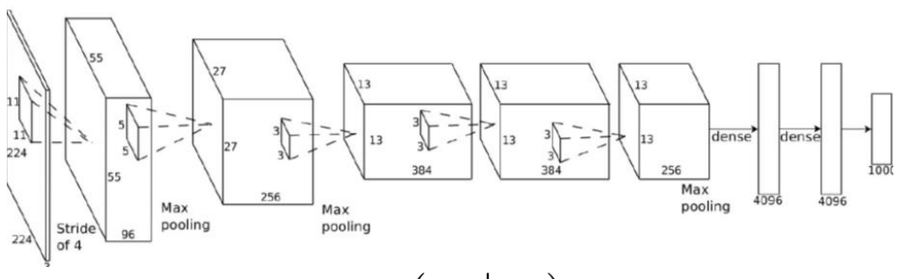
A Reduction of Imitation Learning and Structured Prediction to No-Regret Online Learning, Ross et al. 2011

What can go wrong?

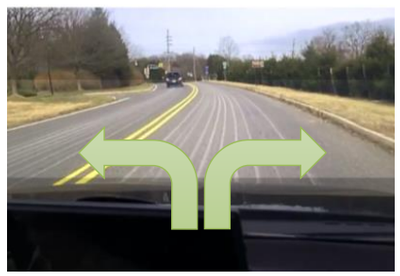
- Compounding errors
  - Fix: data augmentation
- Stochastic expert actions
  - Fix: stochastic latent variable models, action discretiation, gaussian mixture networks
- Non-Markovian observations



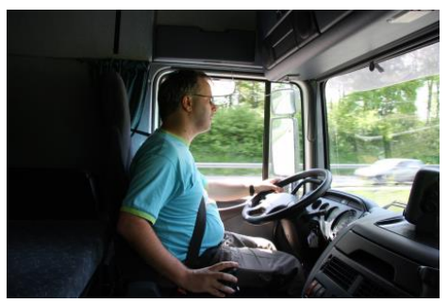
$\mathbf{o}_t$



$\pi_\theta(\mathbf{u}_t|\mathbf{o}_t)$

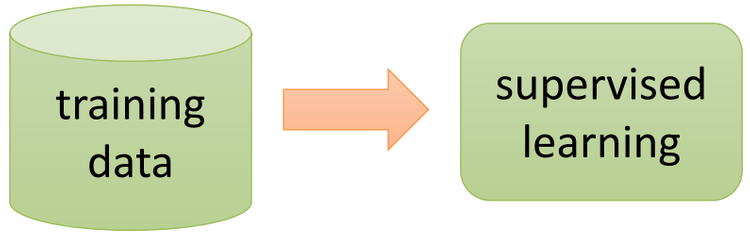


$\mathbf{u}_t$



$\mathbf{o}_t$

$\mathbf{u}_t$



$\pi_\theta(\mathbf{u}_t|\mathbf{o}_t)$

End to End Learning for Self-Driving Cars, Bojarski et al. 2016

Markov property

A stochastic process has the Markov property if the conditional probability distribution of future states of the process (conditional on both past and present states) depends only upon the present state, not on the sequence of events that preceded it

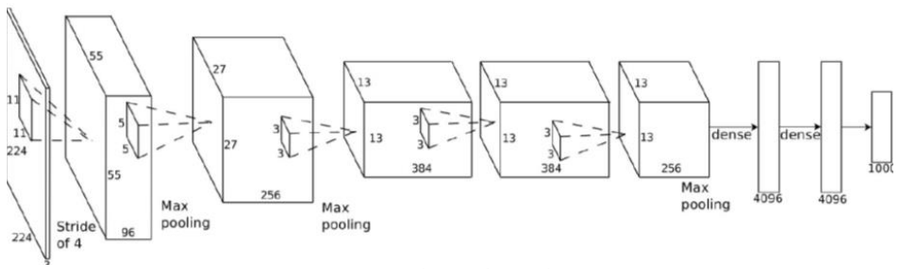
$$\mathbb{P}[R_{t+1} = r, S_{t+1} = s' | S_0, A_0, R_1, ..., S_{t-1}, A_{t-1}, R_t, S_t, A_t] = \mathbb{P}[R_{t+1} = r, S_{t+1} = s' | S_t, A_t]$$

for all $s' \in \mathcal{S}, r \in \mathcal{R}$, and all histories

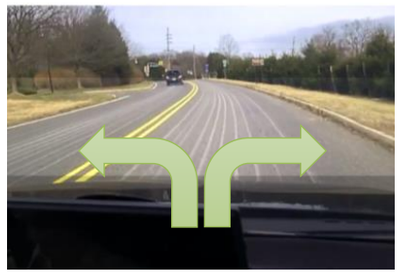
What can go wrong?

- Compounding errors
  Fix: data augmentation
- Stochastic expert actions
  Fix: stochastic latent variable models, action discretiation, gaussian mixture networks
- Non-Markovian observations
  Fix: observation concatenation or recurrent models

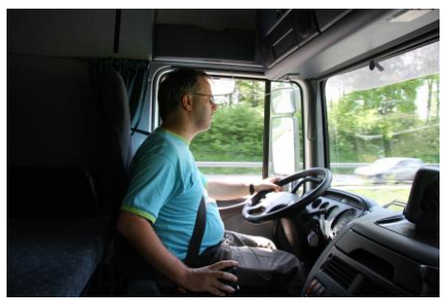
$\mathbf{o}_t$



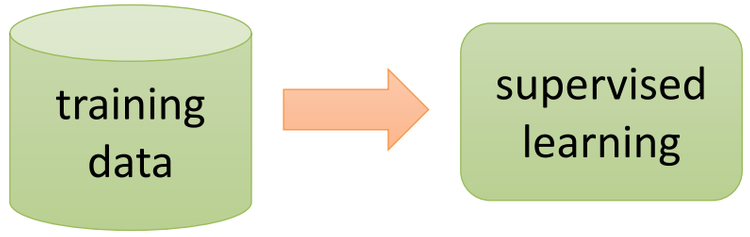
$\pi_\theta(\mathbf{u}_t|\mathbf{o}_t)$



$\mathbf{u}_t$



$\mathbf{o}_t$
$\mathbf{u}_t$

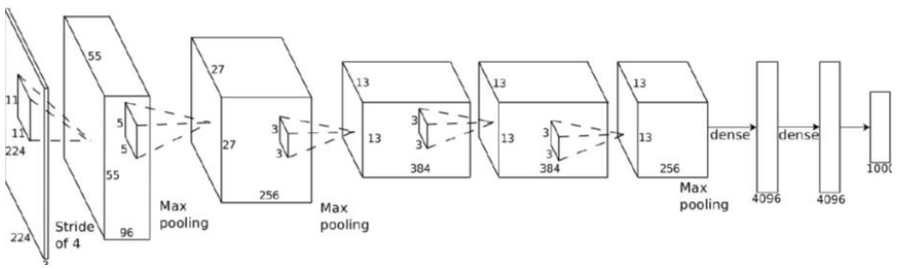


$\pi_\theta(\mathbf{u}_t|\mathbf{o}_t)$

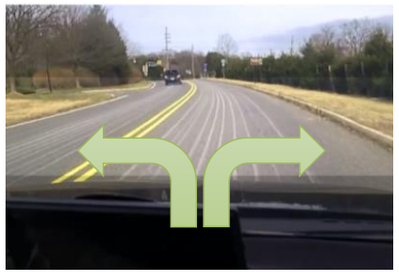
End to End Learning for Self-Driving Cars, Bojarski et al. 2016

Non-markovian observations

$\mathbf{o}_t$



$\pi_\theta(\mathbf{u}_t|\mathbf{o}_t)$



$\mathbf{u}_t$

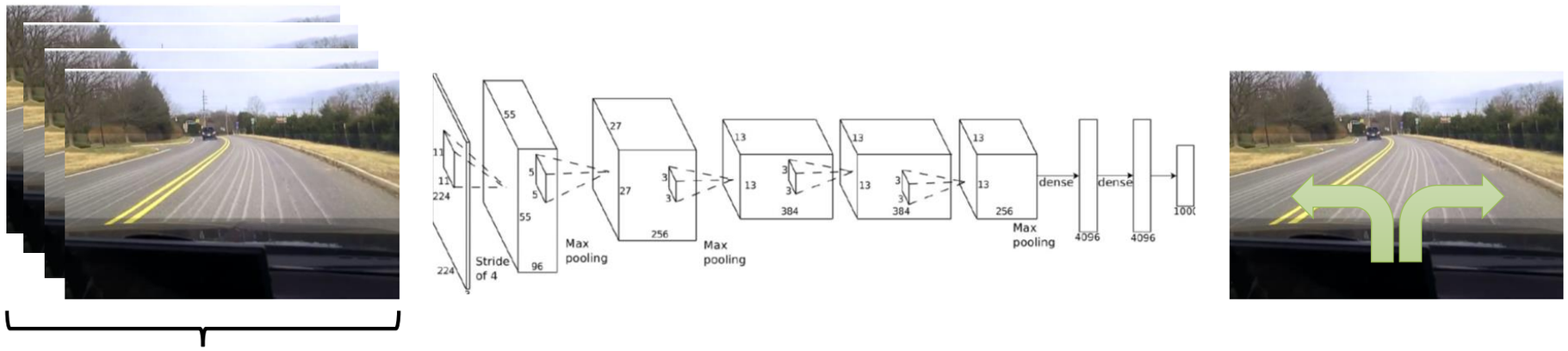
$\pi_\theta(\mathbf{u}_t|\mathbf{o}_t)$

behavior depends only
on current observation

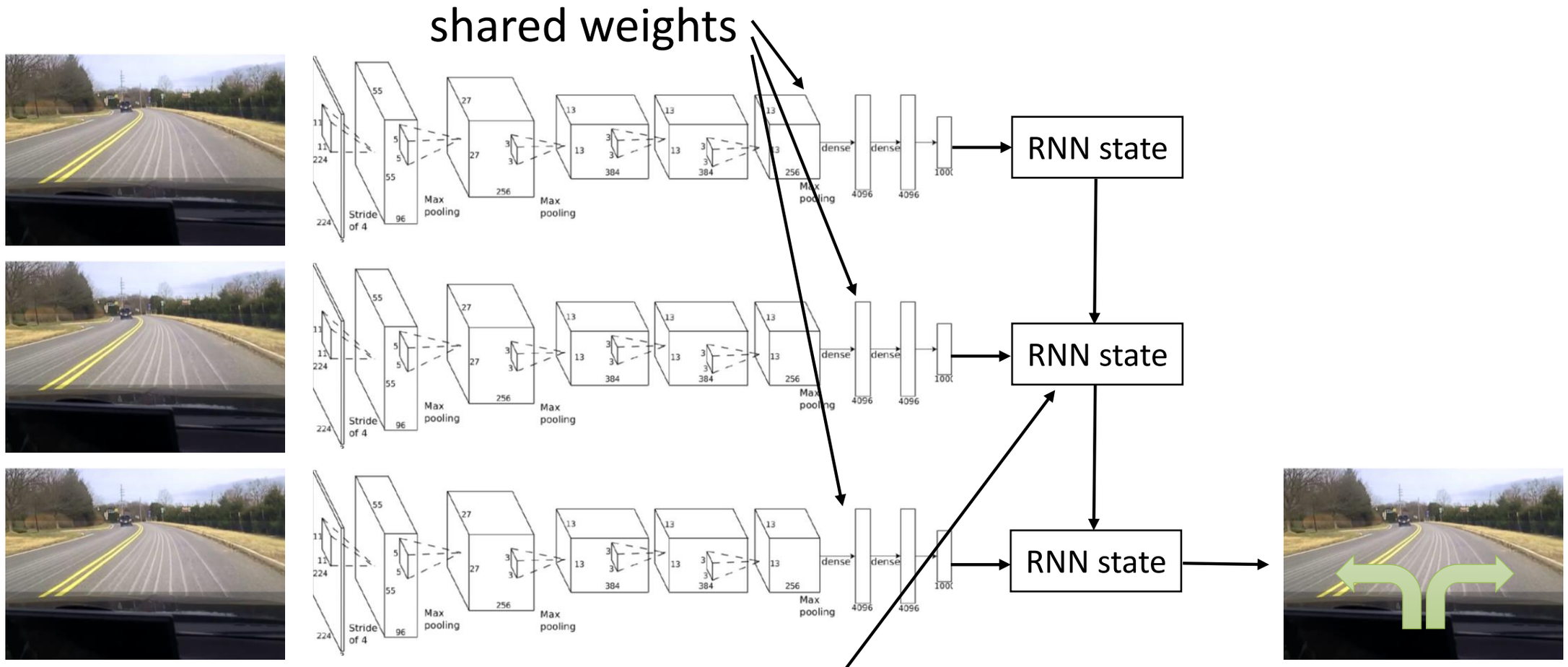
$\pi_\theta(\mathbf{u}_t|\mathbf{o}_1, \ldots, \mathbf{o}_t)$

behavior depends on
all past observations

Fix 1: concatenate observations



Fix 2: use recurrent networks



Typically, LSTM cells work better here

Diagram from Sergey Levine

Recurrent Neural Networks (RNNs)

- RNNs tie the weights at each time step
- Condition the neural network on all previous inputs
- In principle, any interdependencies can be modeled across time steps.
- In practice, limitations from SGD training, capacity, initialization etc.

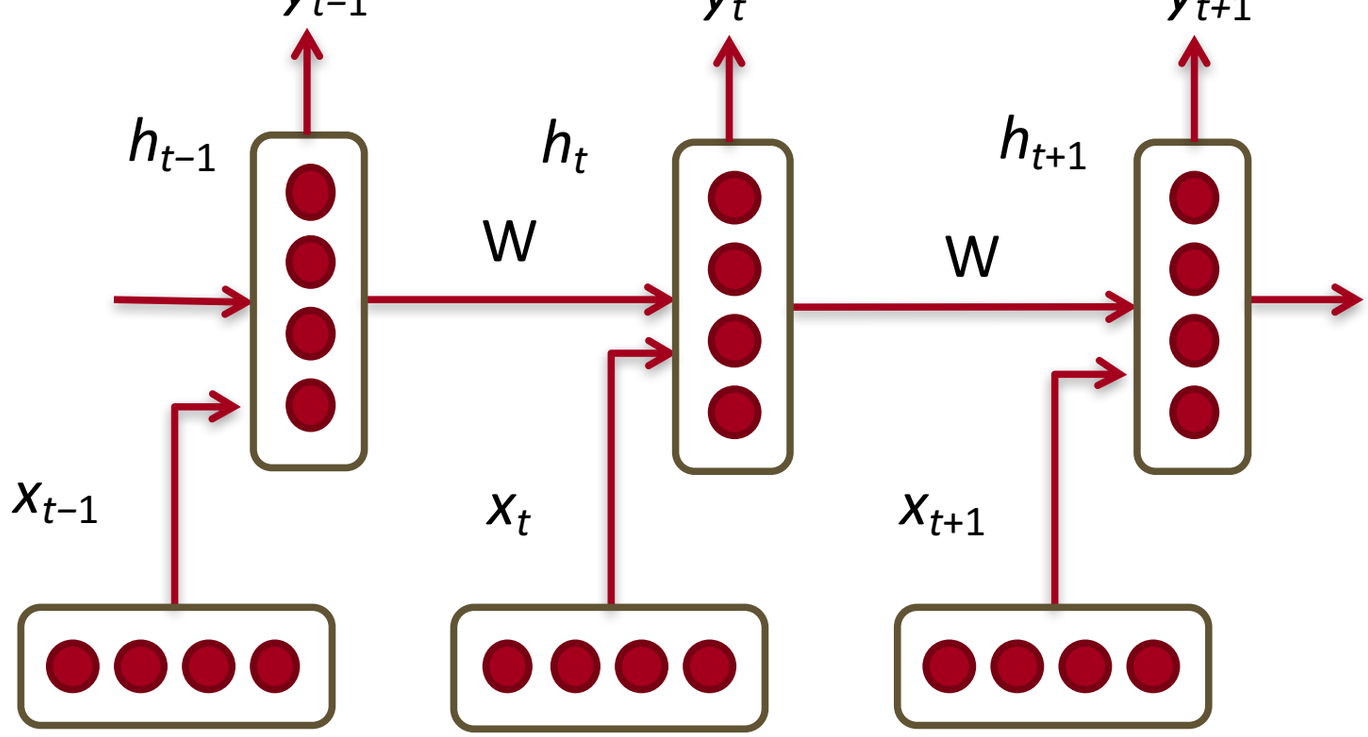


Diagram from Richard Socher

Recurrent Neural Network (single hidden layer)

- Given list of **vectors**:
- At a single time step:

$$x_1, \ldots, x_{t-1}, x_t, x_{t+1}, \ldots, x_T$$

$$h_t = \sigma\left(W^{(hh)} h_{t-1} + W^{(hx)} x_{[t]}\right)$$

$$\hat{y}_t = \text{softmax}\left(W^{(S)} h_t\right)$$

(in case of discrete labels)

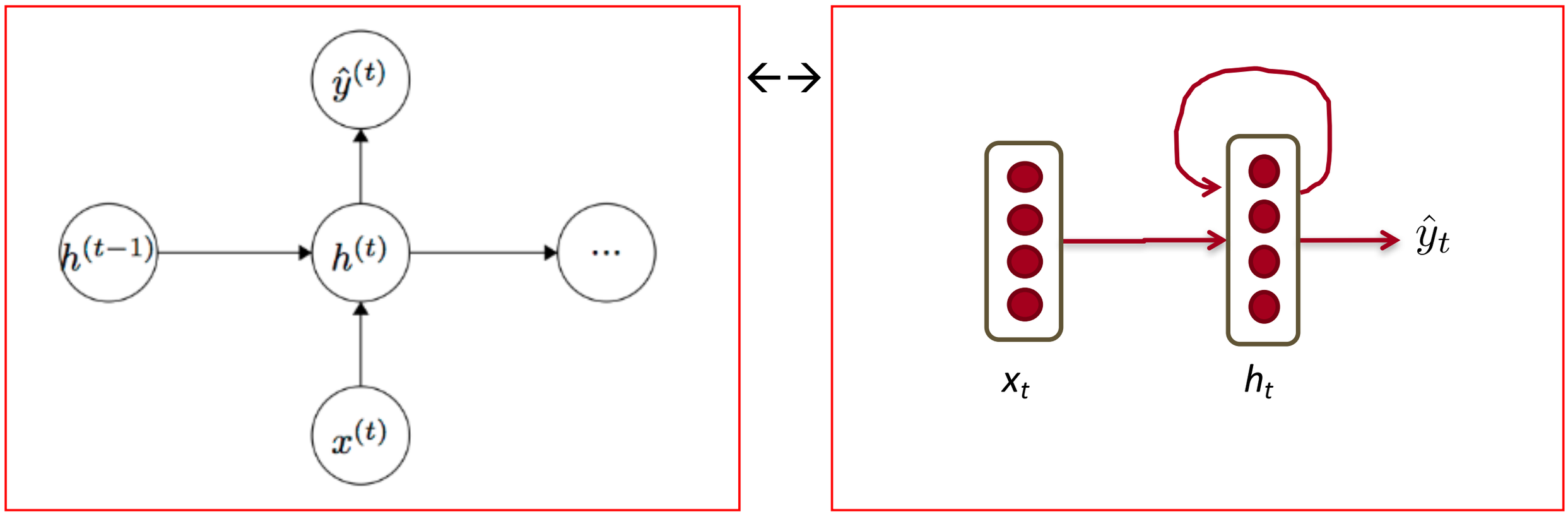
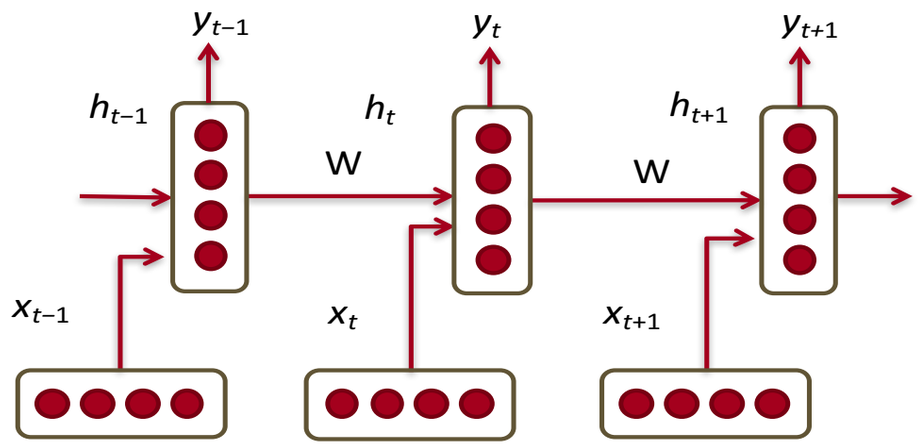


Diagram from Richard Socher

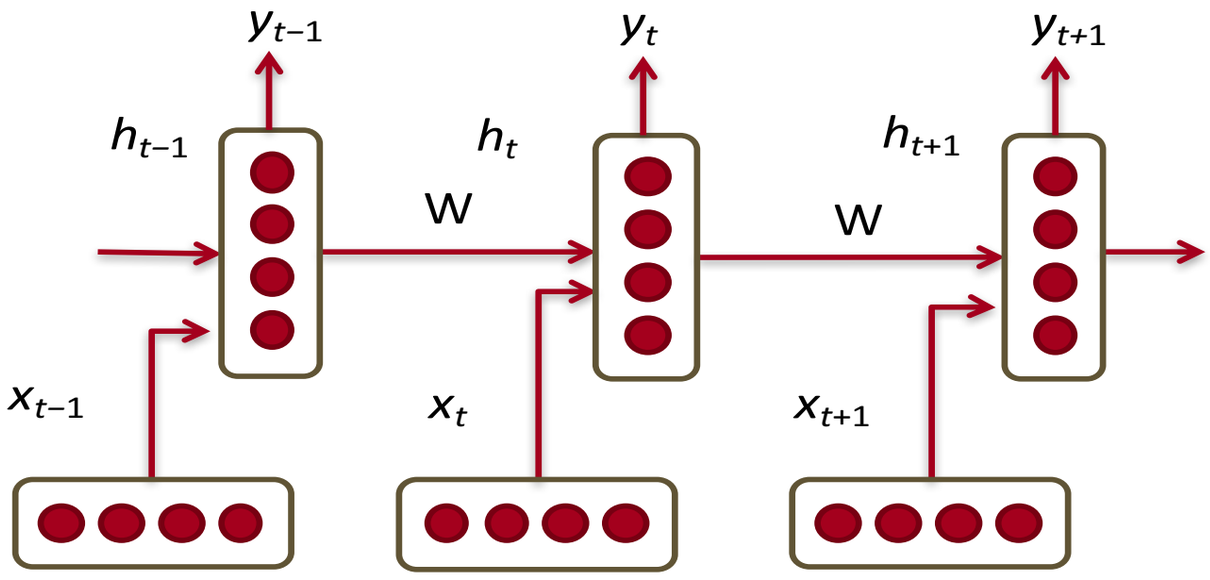
Recurrent Neural Networks

For sequence labelling problems, actions of the labelling policies are y_t, e.g., part of speech tags



For sequence generation, actions of the labelling policies are $y_t = x_{t+1}$, e.g., word in answer generation

$$\hat{P}(x_{t+1} = v_j | x_t, ..., x_1) = \hat{y}_{t,j}$$



Fix 2: use recurrent networks

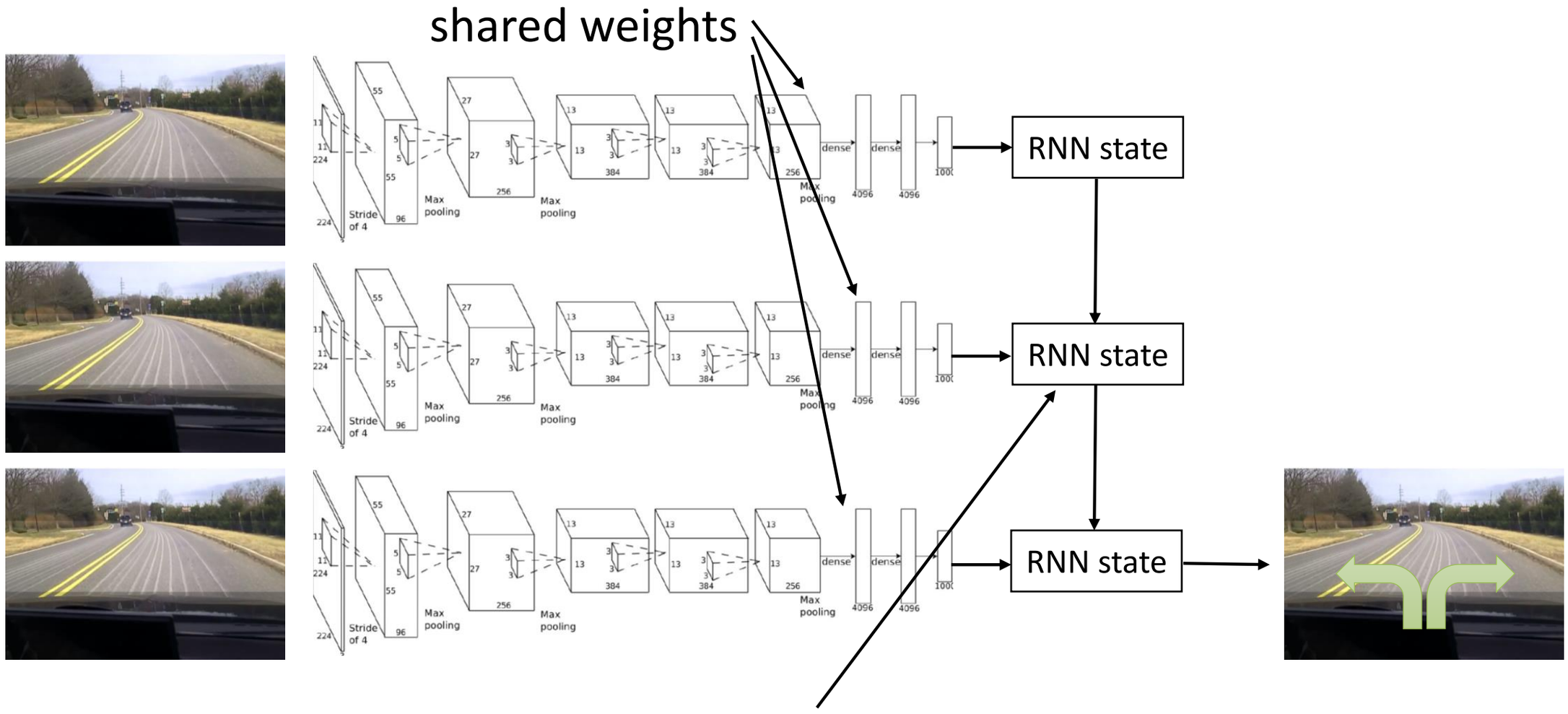


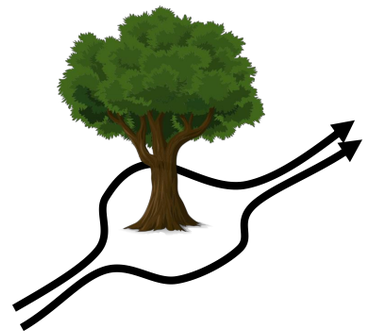
Diagram rom Sergey Levine

Fix 2: use recurrent networks

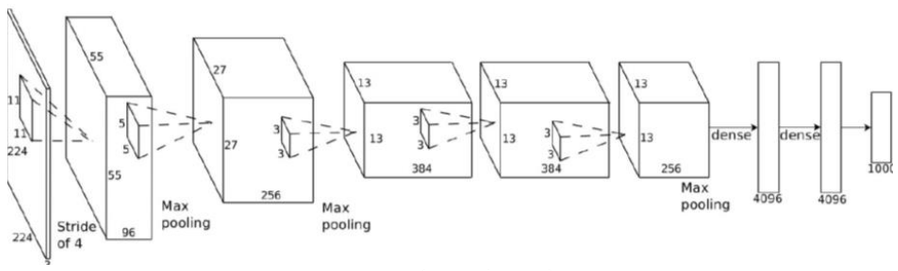
- Usually much more structure is needed in the latent state than a vanilla LSTM can provide, e.g., detections and trackless of objects.
- We will discuss later in the course structured recurrent neural networks for video perception.

What can go wrong?

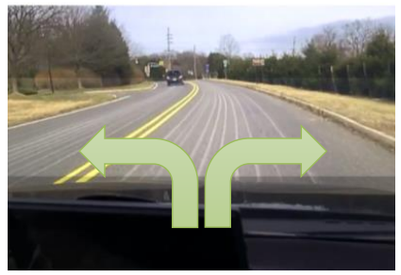
- Compounding errors
  - Fix: data augmentation
- Stochastic expert actions
  - Fix: stochastic latent variable models, action discretiation, gaussian mixture networks
- Non-markovian observations
  - Fix: observation concatenation or recurrent models



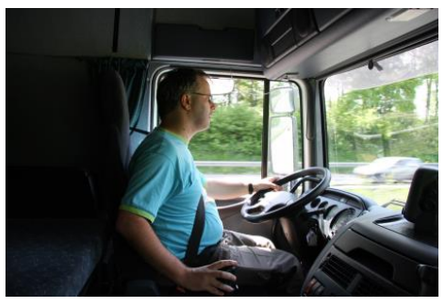
$\mathbf{o}_t$



$\pi_\theta(\mathbf{u}_t|\mathbf{o}_t)$

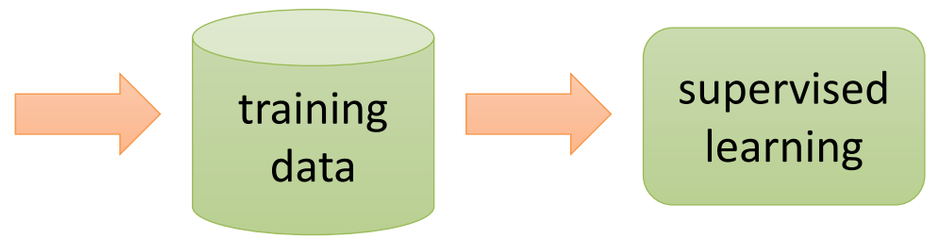


$\mathbf{u}_t$



$\mathbf{o}_t$

$\mathbf{u}_t$

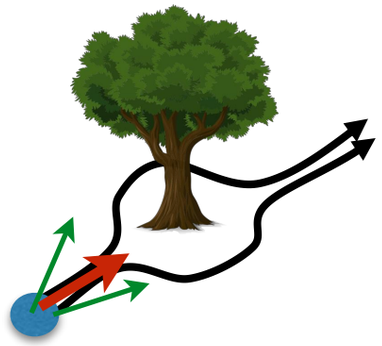


$\pi_\theta(\mathbf{u}_t|\mathbf{o}_t)$

End to End Learning for Self-Driving Cars, Bojarski et al. 2016

Regression fails under multimodality

The answer that minimizes the mean square error is the average which is not a valid prediction



groundtruth streering angles

predicted streering angles

Stochastic expert actions: Fixes

- Discretize the action space and use a classifier (e.g., softmax output and cross-entropy loss)
- Use gaussian mixture model as an output layer, mixture components weights, means and variances are parametrized at the output of a neural net, minimize GMM loss, (e.g., Handwriting generation Graves 2013)
- Stochastic neural networks (later lecture)

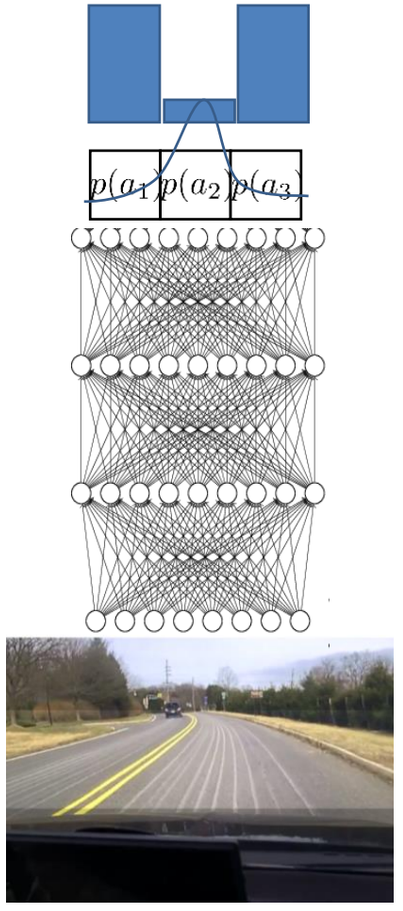
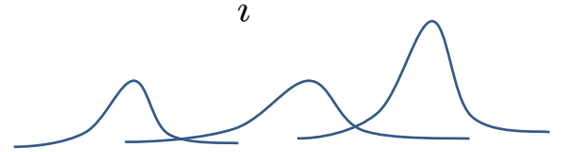


Diagram from Sergey Levine

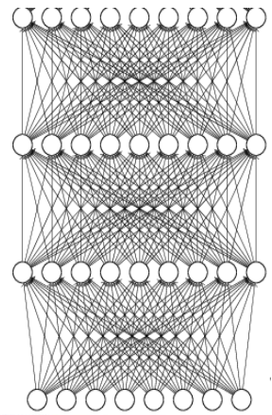
Stochastic expert actions: Fixes

- Discretize the action space and use a classifier (e.g., softmax output and cross-entropy loss)
- Use gaussian mixture model as an output layer, mixture components weights, means and variances are parametrized at the output of a neural net, minimize GMM loss, (e.g., Handwriting generation Graves 2013)
- Stochastic neural networks (later lecture)

$$\pi(\mathbf{a}|\mathbf{o}) = \sum_i w_i \mathcal{N}(\mu_i, \Sigma_i)$$



$$w_1, \mu_1, \Sigma_1, \ldots, w_N, \mu_N, \sigma_N$$



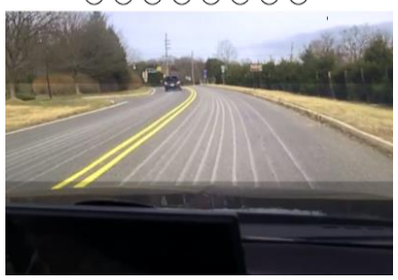


Diagram from Sergey Levine

Stochastic expert actions: Fixes

- Discretize the action space and use a classifier (e.g., softmax output and cross-entropy loss)
- Use gaussian mixture model as an output layer, mixture components weights, means and variances are parametrized at the output of a neural net, minimize GMM loss, (e.g., Handwriting generation Graves 2013)
- Stochastic neural networks (later lecture)

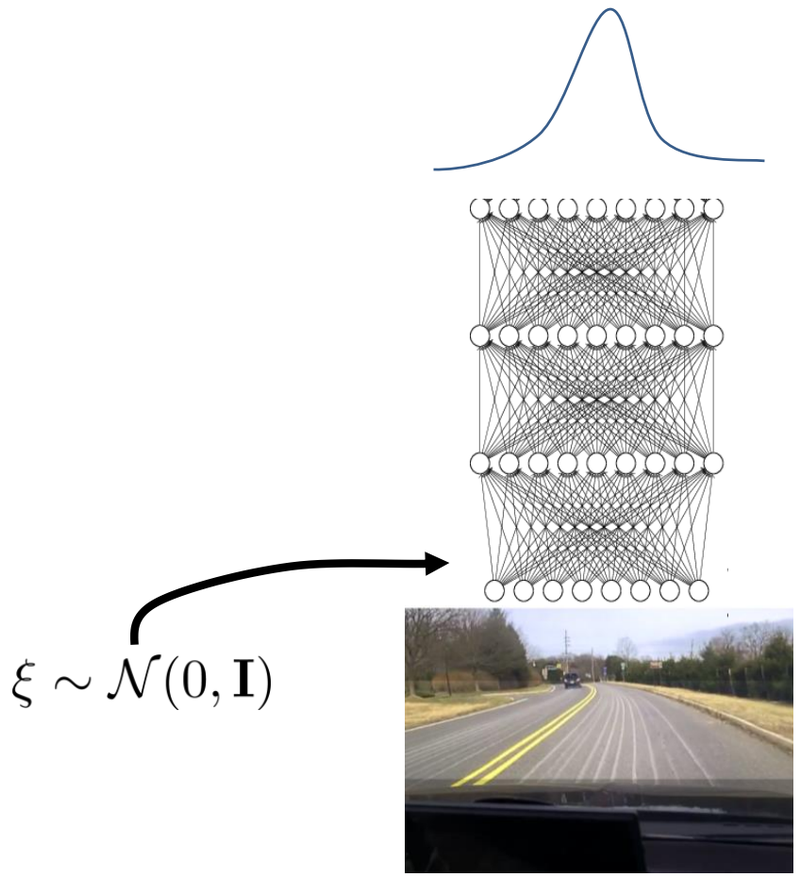


Diagram from Sergey Levine

Structured prediction

Structured prediction: a learner makes predictions over a set of interdependent output variables and observes a joint loss.

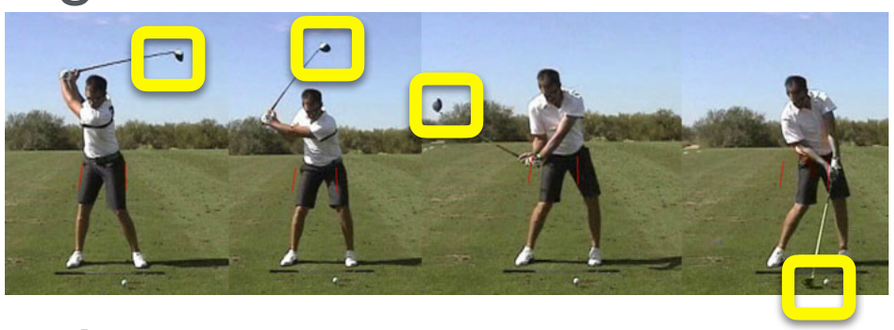
part-of-speech tagging

```
x = the monster ate the sandwich
y = Dt    Nn    Vb  Dt      Nn
```

NER (Name Entity Recognition)

```
x = Yesterday I traveled to Lille
y =     -    PER   -       -   LOC
```

tracking



captioning

“A blue monster is eating a cookie”

Machine translation

Few images from Hall Daume III

Recurrent Neural Networks

The regular training procedure of RNNs treat true labels y_t as actions while making forward passes. Hence, the learning agent follows trajectories generated by the reference policy rather than the learned policy. In other words, it learns:

$$\hat{\theta} = \arg\min_{\theta} \mathbb{E}_{h \sim d_{\pi^*}} [l_\theta(h)]$$

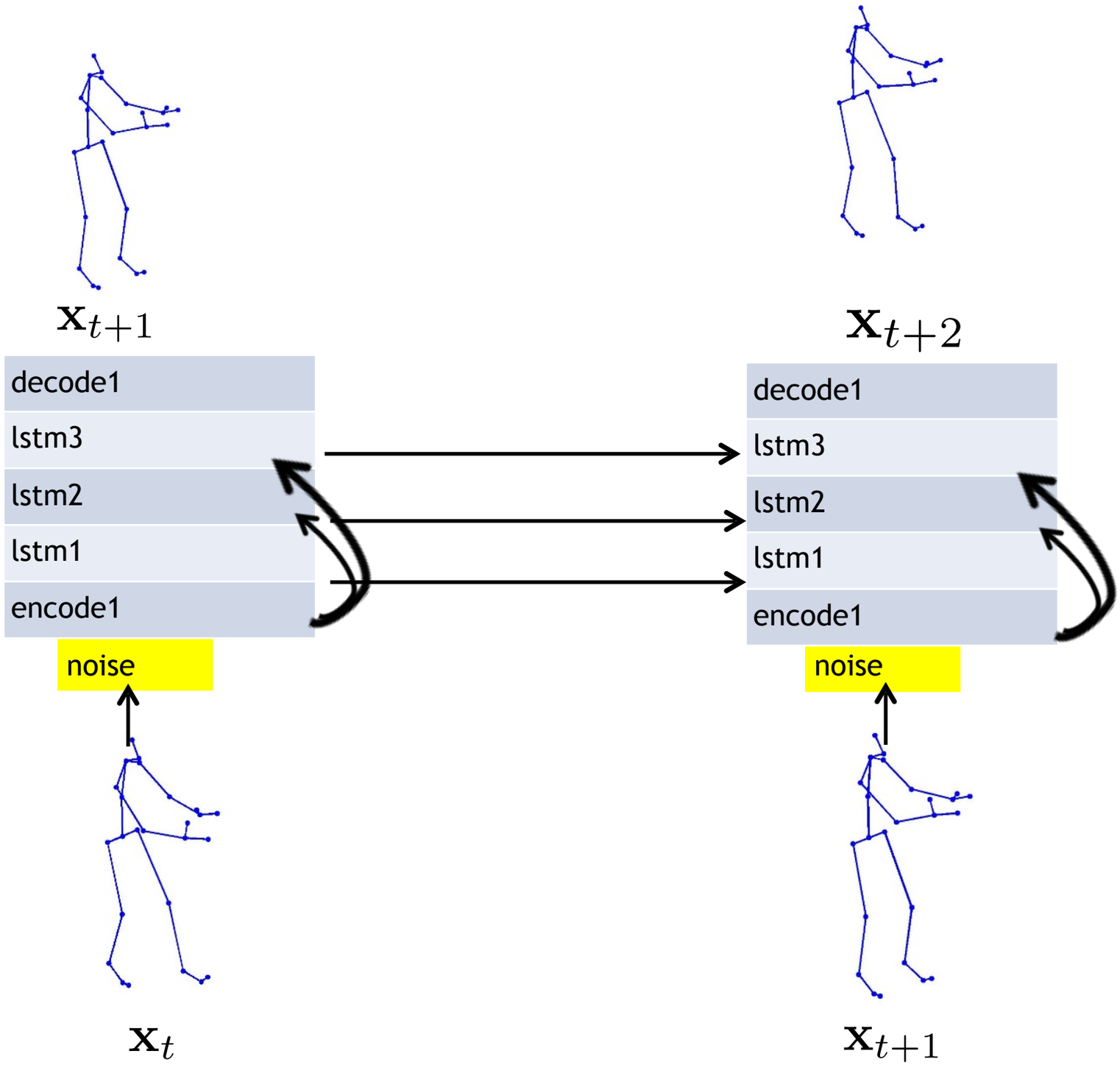
However, our true goal is to learn a policy that minimizes error under its own induced state distribution:

$$\theta^{sup} = \arg\min_{\theta} \mathbb{E}_{h \sim d_{\pi^*}} [l_\theta(h)]$$

$$\hat{\theta} = \arg\min_{\theta} \mathbb{E}_{h \sim d_\theta} [l_\theta(h)]$$

Imitation Learning with Recurrent Neural Networks, Nyuyen 2016

Mocap generation



Scheduled sampling for sequence labelling/generation

Q: what we be feeding the groundtruth x,y or the predicted x,y during training?

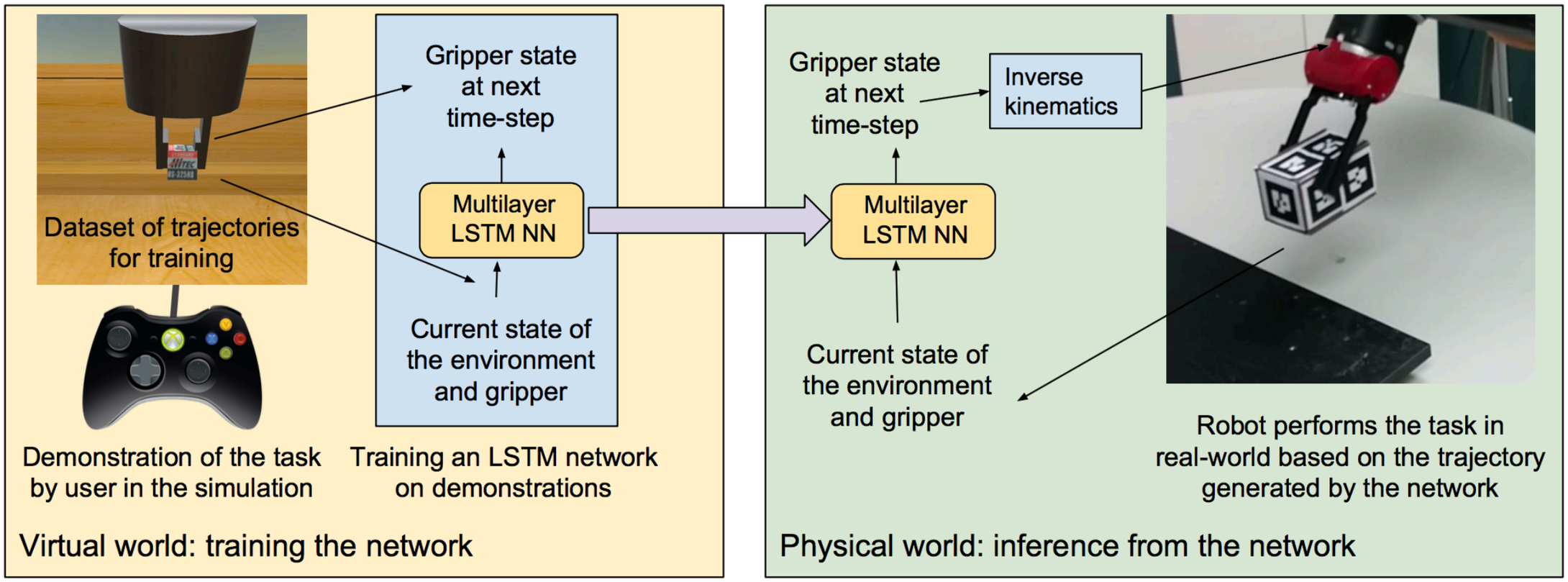
1: **function** TRAIN(N, α)
2: Intialize $\alpha = 1$.
3: Initialize model parameters θ.
4: **for** $i = 1..N$ **do**
5: Set $\alpha = \alpha \cdot p$.
6: Randomize a batch of labeled examples.
7: **for** each example (x, y) in the batch **do**
8: Initialize $h_0 = \Phi(X)$.
9: Initialize $\mathcal{D} = \{(h_0, y_0)\}$.
10: **for** $t = 1 \ldots |Y|$ **do**
11: Uniformly randomize a floating-number $\beta \in [0, 1)$.
12: **if** $\alpha < \beta$ **then**
13: Use true label $\tilde{y}_{t-1} = y_{t-1}$
14: **else**
15: Use predicted label: $\tilde{y}_{t-1} = \arg\max_y P(y \mid h_{t-1}; \theta)$.
16: **end if**
17: Compute the next state: $h_t = f_\theta(h_{t-1}, \tilde{y}_{t-1})$.
18: Add example: $\mathcal{D} = \mathcal{D} \cup \{(h_t, y_t)\}$.
19: **end for**
20: **end for**
21: Online update θ by $\mathcal{D}$ (mini-batch back-propagation).
22: **end for**
23: **end function**

Teacher forcing

Scheduled Sampling for Sequence Prediction with Recurrent Neural Networks, Bengio(Samy) et al.

Imitation Learning with Recurrent Neural Networks, Nyuyen

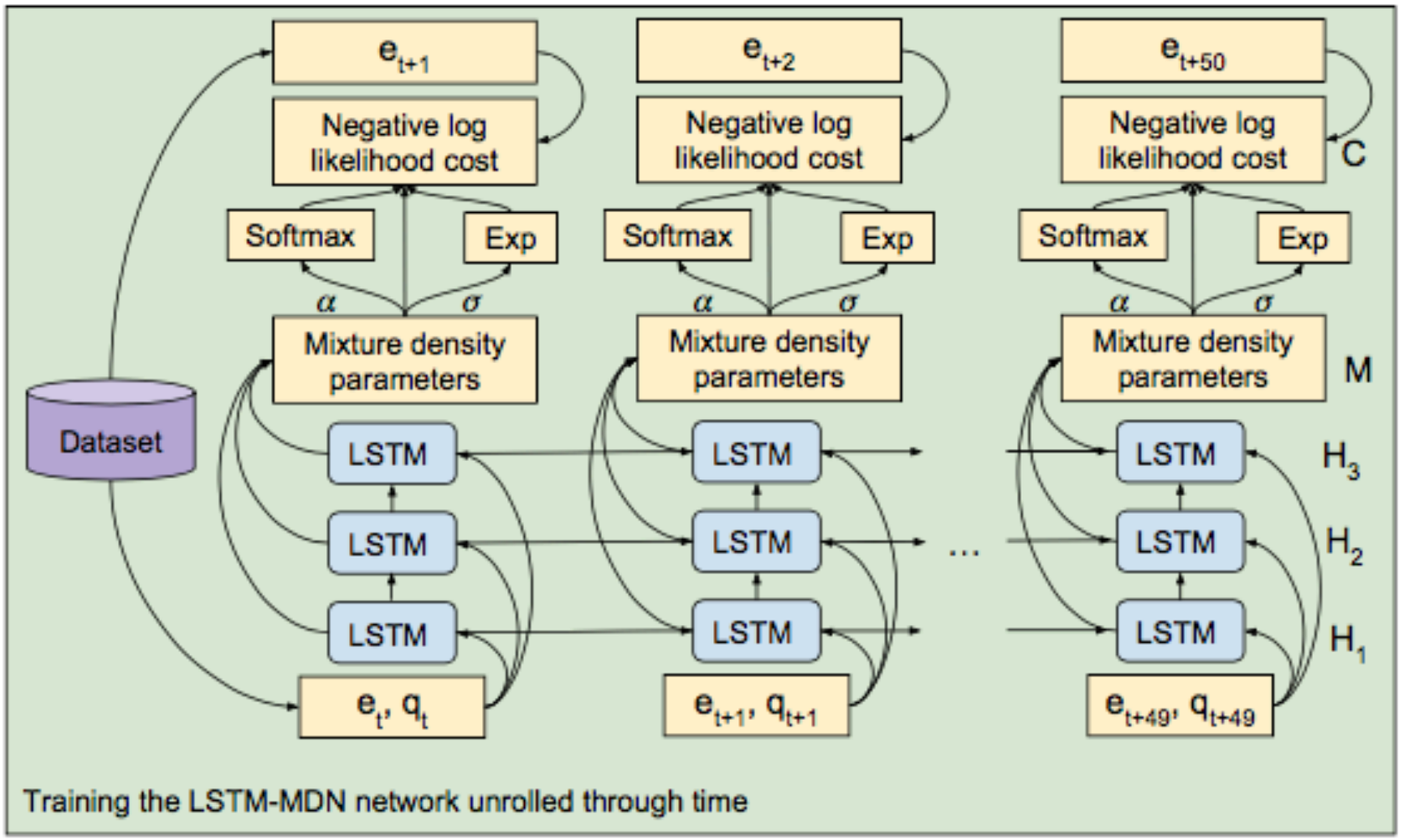
Case study: learning from virtual demonstrations



- Two tasks considered: pick and place, move to desired pose
- State representation x: the poses of all the objects in the seen (rotations, translations) and the pose of the end effector
- Output y: the desired next pose of the end effector
- Supervision: expert trajectories in the simulator
- Demonstation augmentation: consider multiple trajectories by subsampling in time the expert ones, and by translating in space the end effector

Learning real manipulation tasks from virtual demonstrations using LSTM, Rahmatizadeh et al 2016

Case study: learning from virtual demonstrations

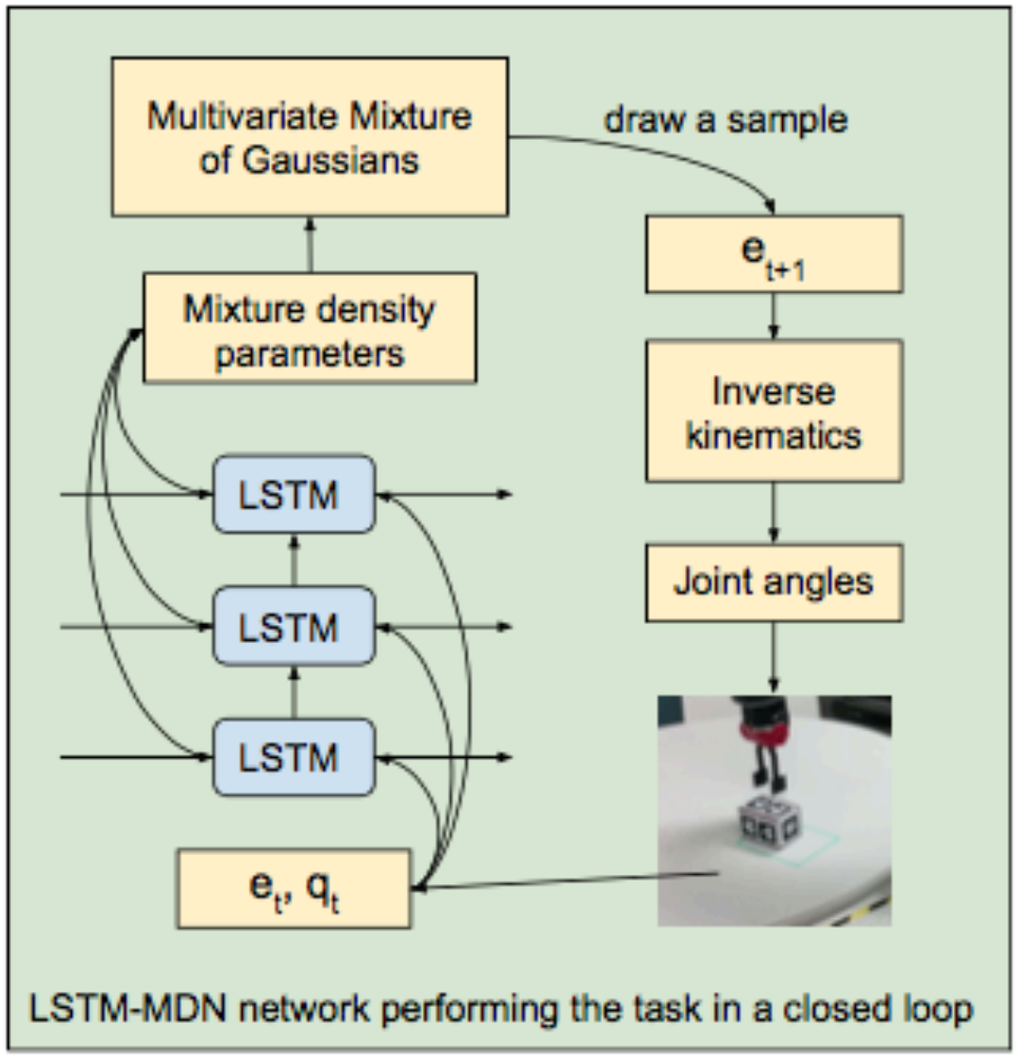


- Multimodality of actions-> GMM loss!

- Predict mixture weights over a Gaussian Mixture Model at the output (alphas) and mean and variances for the mixture components.

Learning real manipulation tasks from virtual demonstrations using LSTM, Rahmatizadeh et al 2016

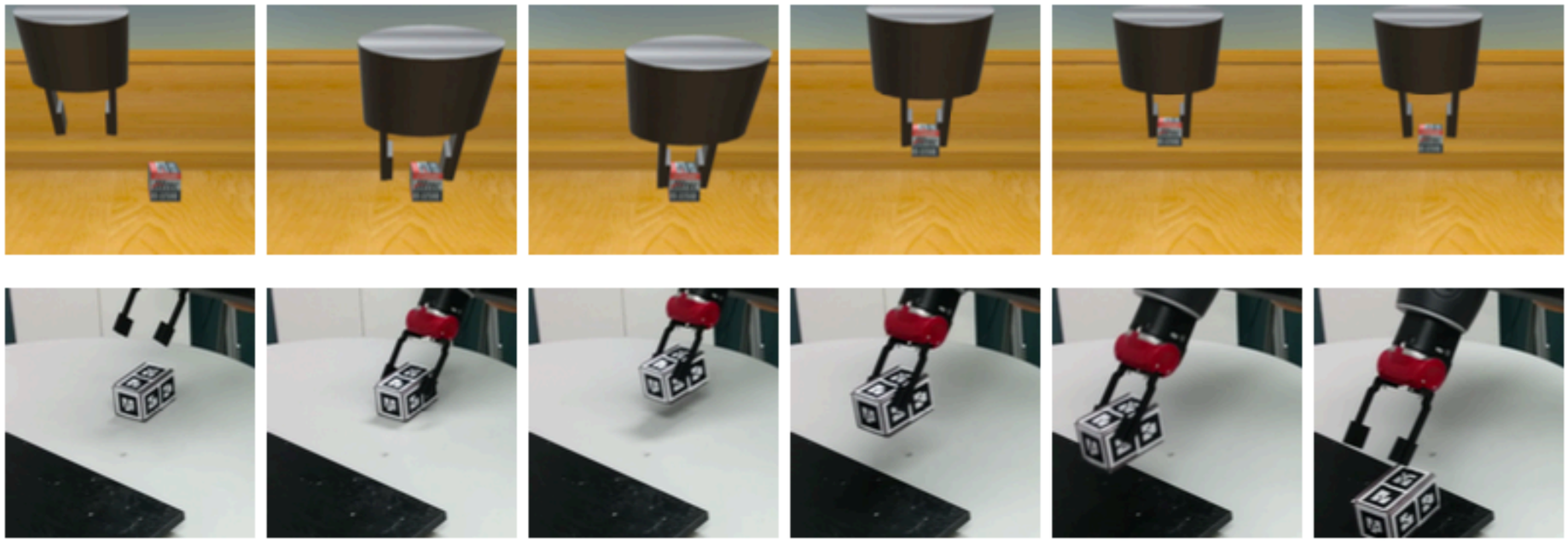
Case study: learning from virtual demonstrations



- Multimodality: predict mixture weights over a Gaussian Mixture Model at the output (alphas) and mean and variances for the mixture components. Minimize a GMM loss.

Learning real manipulation tasks from virtual demonstrations using LSTM, Rahmatizadeh et al 2016

Case study: learning from virtual demonstrations



Learning real manipulation tasks from virtual demonstrations using LSTM, Rahmatizadeh et al 2016

Case study: learning from virtual demonstrations

Learning Manipulation Trajectories

Using Recurrent Neural Networks

Learning real manipulation tasks from virtual demonstrations using LSTM, Rahmatizadeh et al 2016

ChauffeurNet: Learning to Drive by Imitating the Best and Synthesizing the Worst

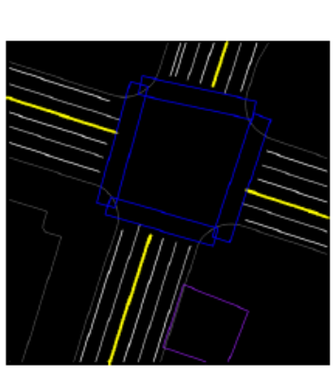
Mayank Bansal MAYBAN@WAYMO.COM
Waymo Research
Mountain View, CA 94043, USA

Alex Krizhevsky* AKRIZHEVSKY@GMAIL.COM

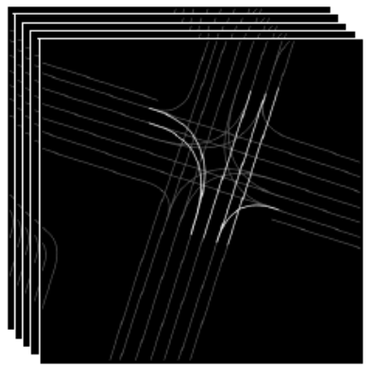
Abhijit Ogale OGALE@WAYMO.COM
Waymo Research
Mountain View, CA 94043, USA

- 30 million examples - 60 days of driving
- structured input from perception, not raw pixels
- augmentation to handle distribution shift (compounding errors)

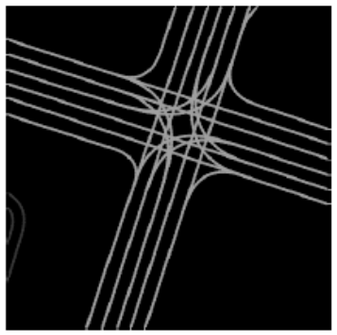
Structured perceptual input



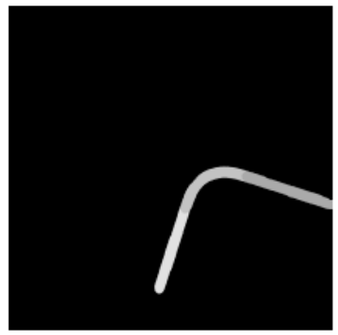
(a) Roadmap



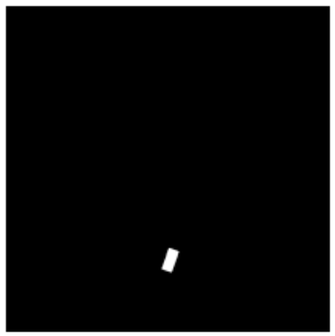
(b) Traffic Lights



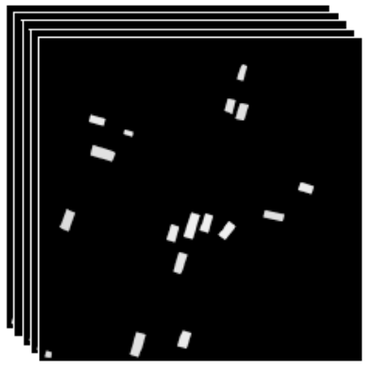
(c) Speed Limit



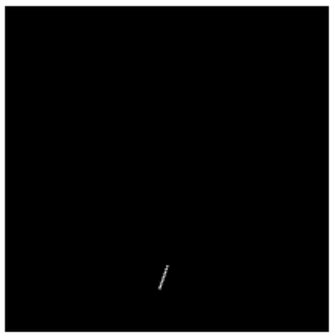
(d) Route



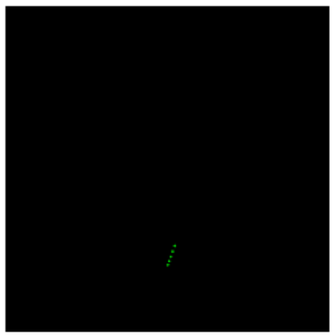
(e) Current Agent Box



(f) Dynamic Boxes



(g) Past Agent Poses



(h) Future Agent Poses

When label maps from multiple time steps are used, the maps are stabilized so that the position of the agent always appears at the same pixel location (ego-stabilization)

Benefits from this structured, abstracted from pixels input:

1) easier to learn from
2) we can easily synthesize data(!) on such abstract format and training the driving network on rare scenarios not frequently seen in the real world.

Car trajectory prediction

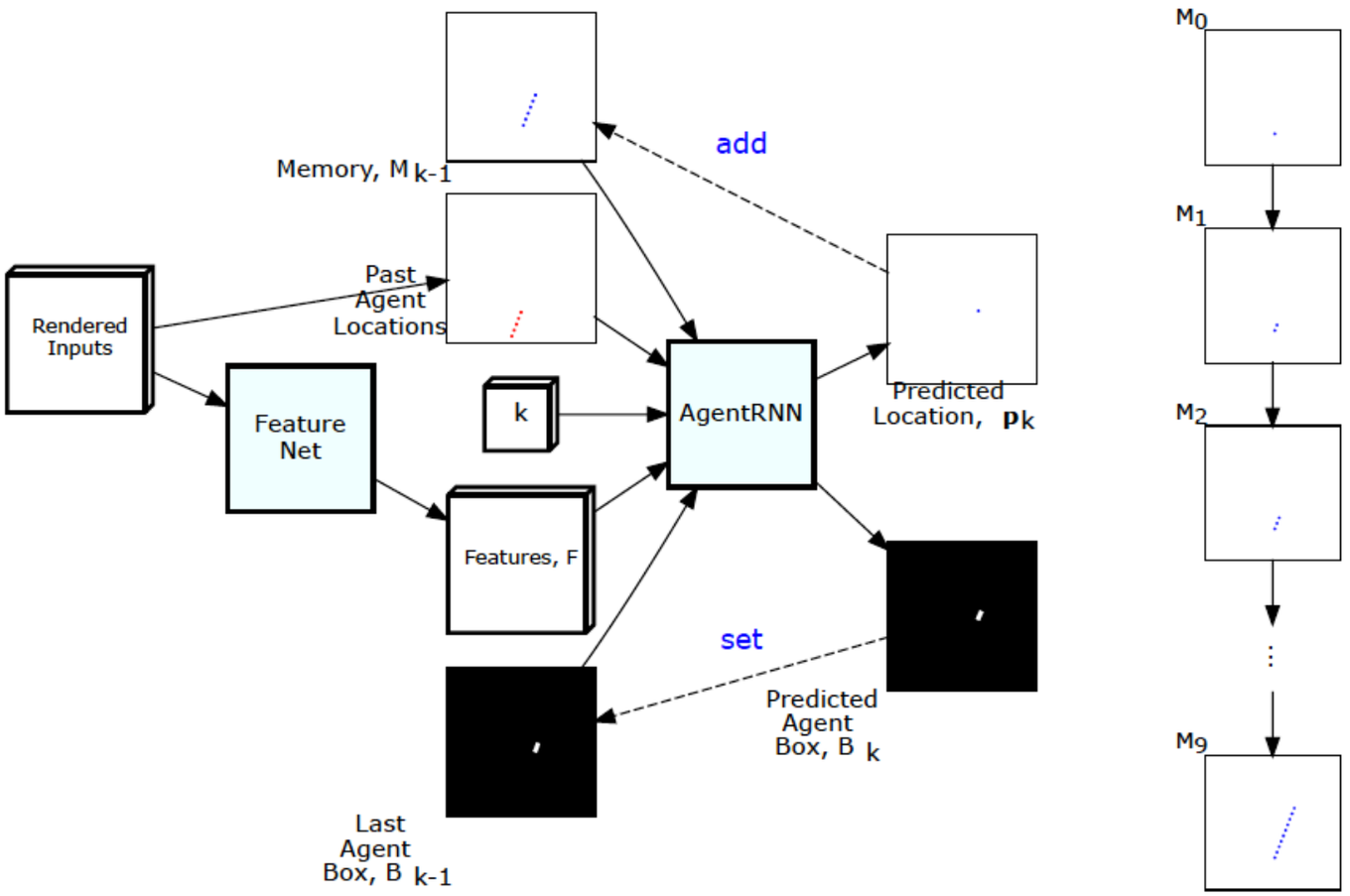


Figure 3: (a) Schematic of ChauffeurNet. (b) Memory updates over multiple iterations.

Q: Why we do not predict steering angle directly, rather, a location (waypoint) + orientation of the car?

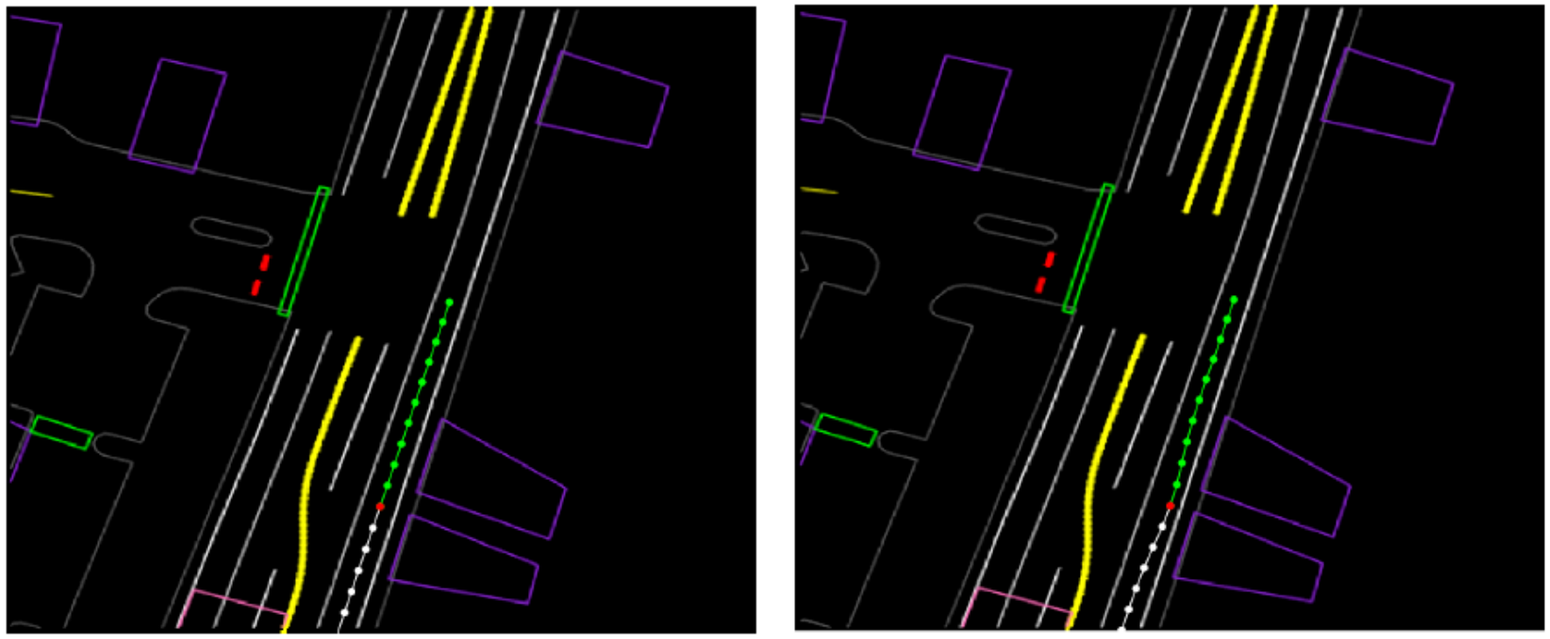
Causal confusion

“During training, the model is provided the past motion history as one of the inputs. Since the past motion history during training is from an expert demonstration, the net can learn to cheat by just extrapolating from the past rather than understanding the underlying causes of the behavior.”

Solution?

(past) motion dropout

Handling distribution shift with synthetic data



(a) Original (b) Perturbed

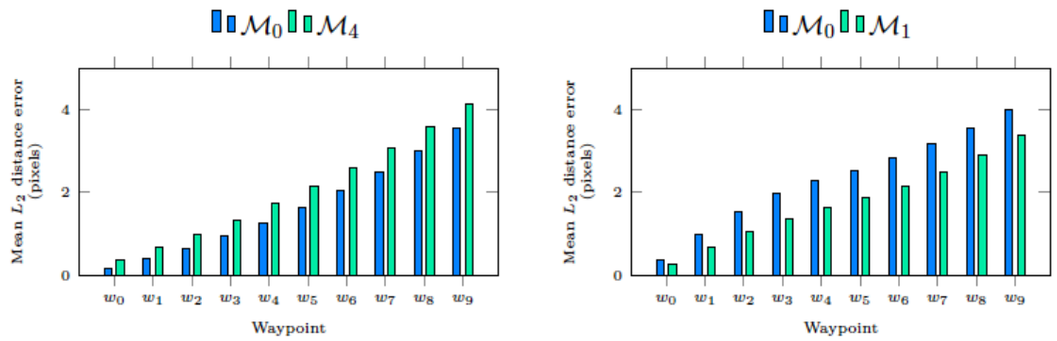
Figure 5: Trajectory Perturbation. (a) An original logged training example where the agent is driving along the center of the lane. (b) The perturbed example created by perturbing the current agent location (red point) in the original example away from the lane center and then fitting a new smooth trajectory that brings the agent back to the original target location along the lane center.

Results

Model	Description	w_{imit}	w_{env}
$\mathcal{M}_0$	Imitation with Past Dropout	1.0	0.0
$\mathcal{M}_1$	$\mathcal{M}_0$ + Traj Perturbation	1.0	0.0
$\mathcal{M}_2$	$\mathcal{M}_1$ + Environment Losses	1.0	1.0
$\mathcal{M}_3$	$\mathcal{M}_2$ with less imitation	0.5	1.0
$\mathcal{M}_4$	$\mathcal{M}_2$ with Imitation Dropout	*Dropout probability = 0.5 (see Section 5.3).*	

Table 3: Model configuration for the model ablation tests.

open loop: isolated state-action pairs



(a) Prediction Error for models $\mathcal{M}_0$ and $\mathcal{M}_4$ on unperturbed evaluation data.

(b) Prediction Error for models $\mathcal{M}_0$ and $\mathcal{M}_1$ on perturbed evaluation data.

Figure 8: Open loop evaluation results.

closed loop: model is used to drive a car

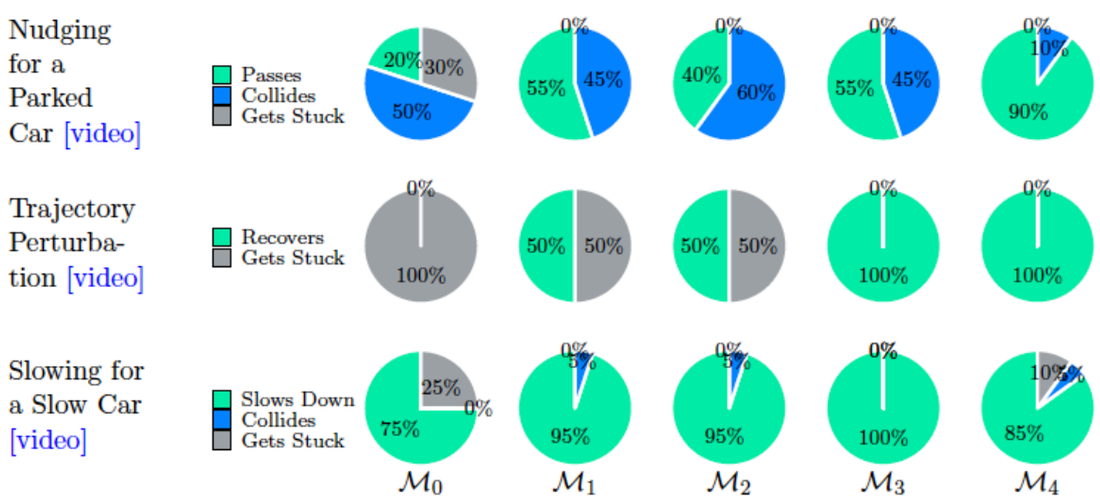


Figure 7: Model ablation test results on three scenario types.

Causal Confusion in Imitation Learning

Pim de Haan **Dinesh Jayaraman** **Sergey Levine**
Berkeley Artificial Intelligence Research, UC Berkeley

Causal Confusion - Image Classification

``hat”

Causal Confusion- Image Classification

``hat”

Causal Confusion- Image Classification

Access to more information makes the problem harder: our model needs to understand what part of the input is related to the label, i.e., causes the label, and which is irrelevant

``hat”

Causal Confusion - Imitation Learning

Assume we have access to the same expert driving trajectories in two setups:

Scenario A

Scenario B

The yellow brake square is an indicator of whether the driver is braking.

Consider training behaviour cloning to imitate the human driver in these two scenarios.

Q1: Do we expect one setup to have lower test error (open loop) than the other?

Q2: Do we expect one setup to learn much better policies for deployment? (closed loop)

Causal Confusion- Imitation Learning

Assume we have access to the same expert driving trajectories in two setups:

Scenario A

Scenario B

It is easy for BC in setup A to obtain low open loop (train and) test error by learning to press the brake whenever the yellow indicator is on.

Q: How will this policy perform during deployment?

Causal confusion

Metrics → / Methods ↓	Validation	Driving Performance		
	Perplexity	Distance	Interventions	Collisions
HISTORY	**0.834**	144.92	2.94 ± 1.79	6.49 ± 5.72
NO-HISTORY	0.989	**268.95**	$\mathbf{1.30 \pm 0.78}$	$\mathbf{3.38 \pm 2.55}$

Table 1: Imitation learning results from Wang et al. [54]. Accessing history yields better validation performance, but worse actual driving performance.

Causal Confusion- Imitation Learning

Assume we have access to the same expert driving trajectories in two setups:

Scenario A

Scenario B

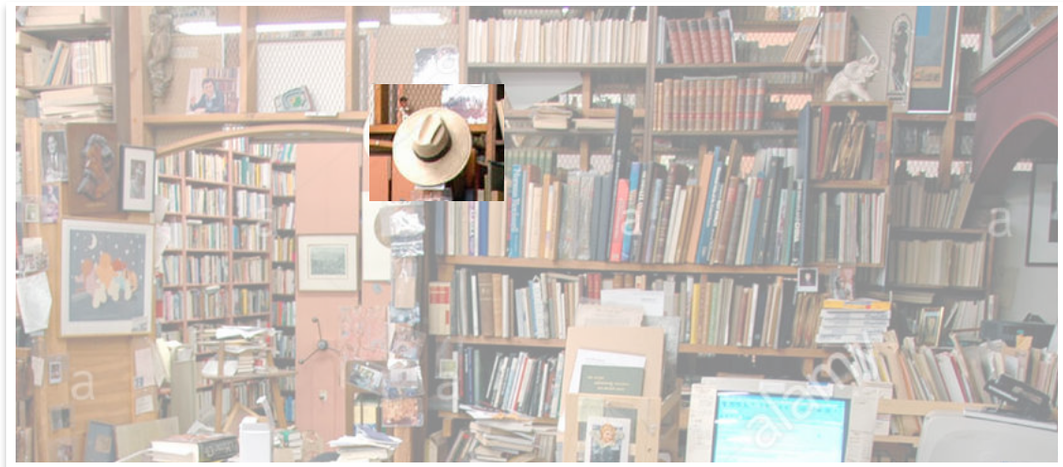
It is easy for BC in setup A to obtain low train and test error by learning to press the brake whenever the yellow indicator is ON. How will this policy perform during deployment?

Q: Would GAIL under setup A solve the problem?

Yes, at a cost of much more interactions with the environment.

Causal Confusion- Imitation Learning

``hat”



- Identifying what are the contributing features for the desired output (driving action or image label) is crucial both for learning good policies and learning good classifiers that generalize. We need to learn to ignore nuisance factors.

Causal Confusion in Imitation Learning,De Haan et al

Causal Confusion - Imitation Learning

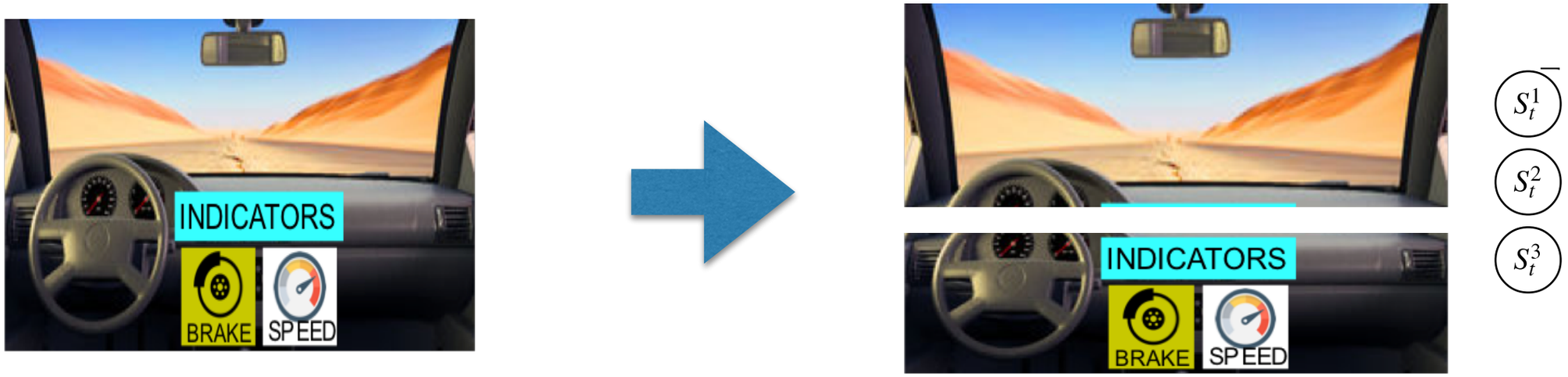
Access to more information makes the problem harder: our model needs to understand what part of the input is related to the label.

``hat”

- In the case of image classification, if you are supply enough data the classifier will eventually learn to attend to the right part, simply because the concept with coincide with diverse nuisance contexts.
- Q: In the case of behaviour cloning (BC), would more data solve the problem?
- because the nuisance factors persist (the dashboard is always there), supplying more data will not help.

Disentangling Observations

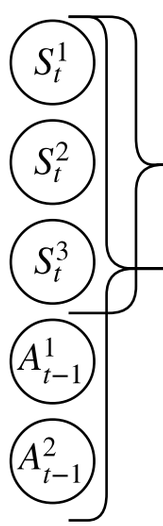
- The **disentangled** state fetures are obtained with a variation on variational autoencoders, $\beta - VAE$



Examples of ideal disentangled visual state features:

- Different objects/parts of the image
- Color - shape - size - distance from the camera of objects

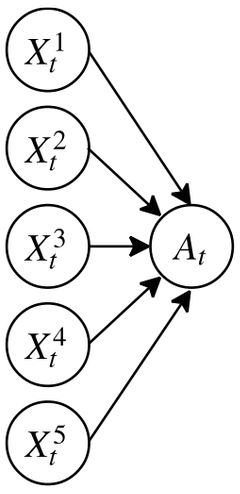
Confounding



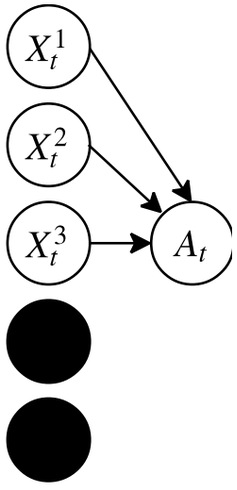
Original state

Confounded state

Task A:



Task B:
True causal graph



- You can easily confuse a behaviour cloning objective by supplying the past actions as part of the current state (used to regress to the next action).
- BC will prooly learn to copy paste the previous actions.
- Q: What is the problem with that?

Causal Confusion in Imitation Learning,De Haan et al

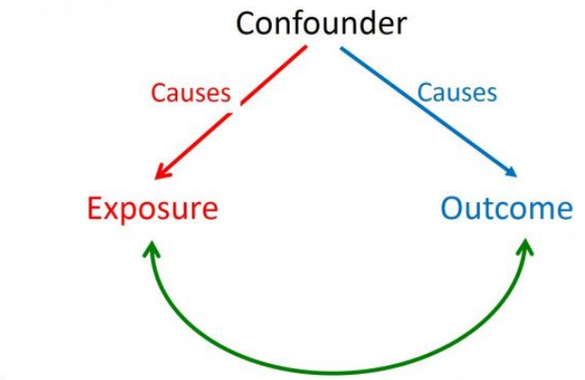
Confounding

- You can easily confuse a behaviour cloning objective by supplying the past actions as part of the current state (used to regress to the next action).
- BC will prooly learn to copy paste the previous actions.
- Q: What is the problem with that?

Causal Confusion in Imitation Learning,De Haan et al

Confounding

Confounder: “an extraneous variable in an experimental design that correlates with both the dependent and independent variables”



AN OLD CLASSIC: MURDER AND ICE CREAM

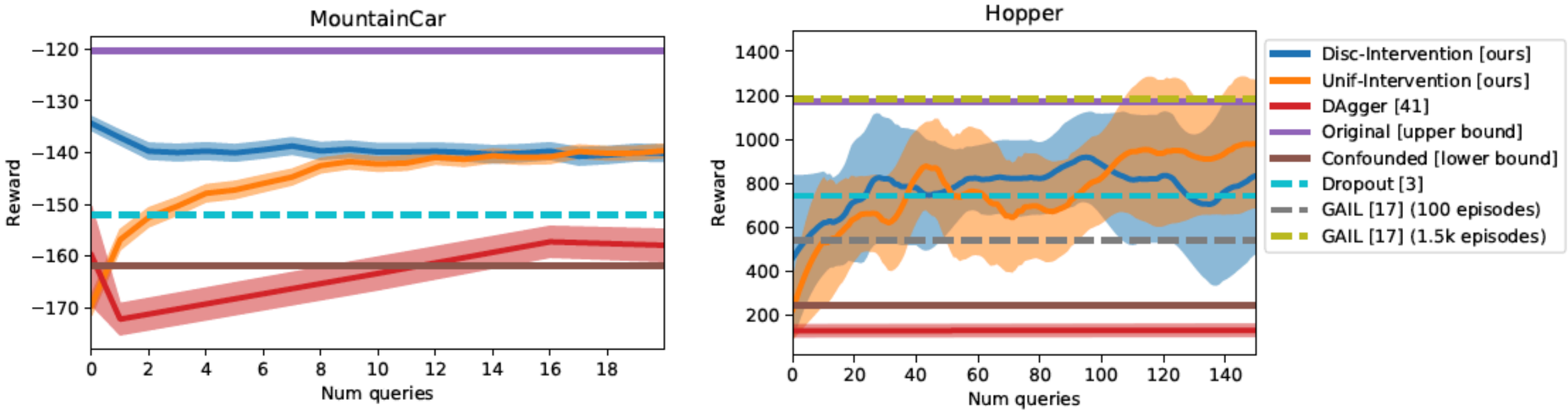
It is known that throughout the year, murder rates and ice cream sales are highly positively correlated. That is, as murder rates rise, so does the sale of ice cream. There are three possible explanations for this correlation:

Possibility #1: Murders cause people to purchase ice cream. One could imagine a world where this is true. Perhaps when one is murdered, they are resurrected as zombies who primarily feed on ice cream.

Possibility #2: Purchasing ice cream causes people to murder or get murdered. Again, one could imagine a world where this is true. Perhaps when one eats ice cream, those without ice cream become jealous and murder those with ice cream.

Possibility #3: There is a third variable—a confounding variable—which causes the increase in BOTH ice cream sales AND murder rates. For instance, the weather. When it’s cold and Wintery, people stay at home rather than go outside and murder people. They also probably don’t eat a lot of ice cream. When it’s hot and Summery, people spend more time outside interacting with each other, and hence are more likely to get into the kinds of situations that lead to murder. They are also probably buying ice cream, because nothing beats the sound of an ice cream truck on a blazing Summer day.

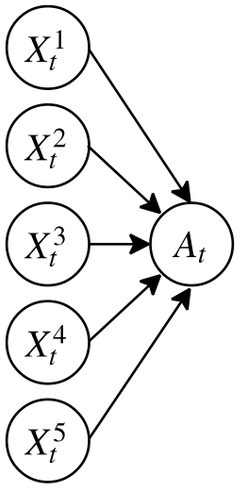
Confounding



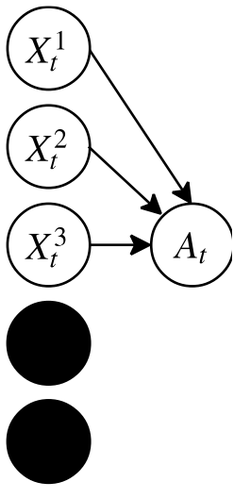
Discovering the causal structure

- Q: Can we come up with a way to discover the right causal structure
- If we have n disentangled factors, each one can be a valid cause or nuisance, then there are 2^n combinations of causes. It is hard to train each one, ask it to drive, and select the one with the best driving performance.

Task A:

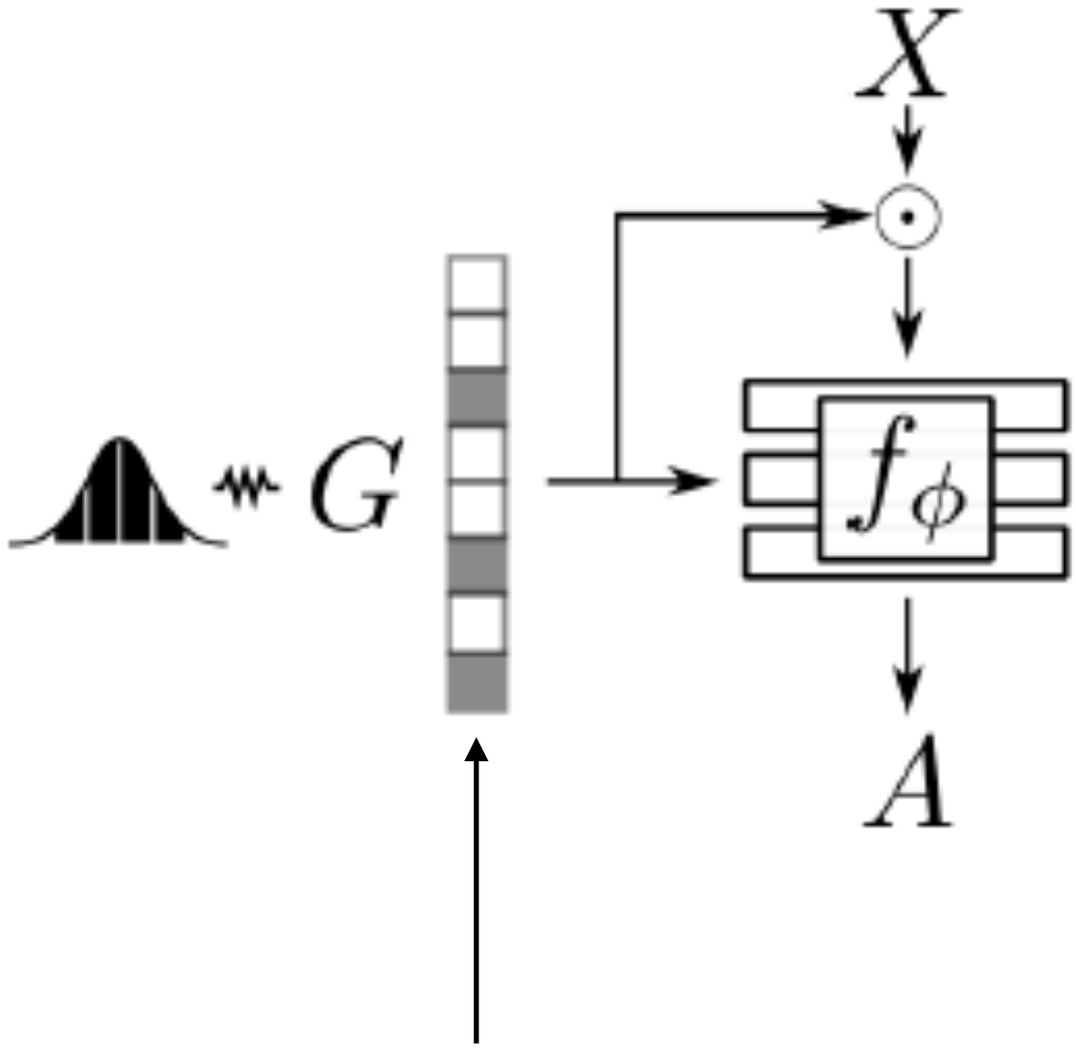


Task B:
True causal graph



Learning a structure-parametrized policy

Learning a mapping from the mask, and the masked state features to the output action



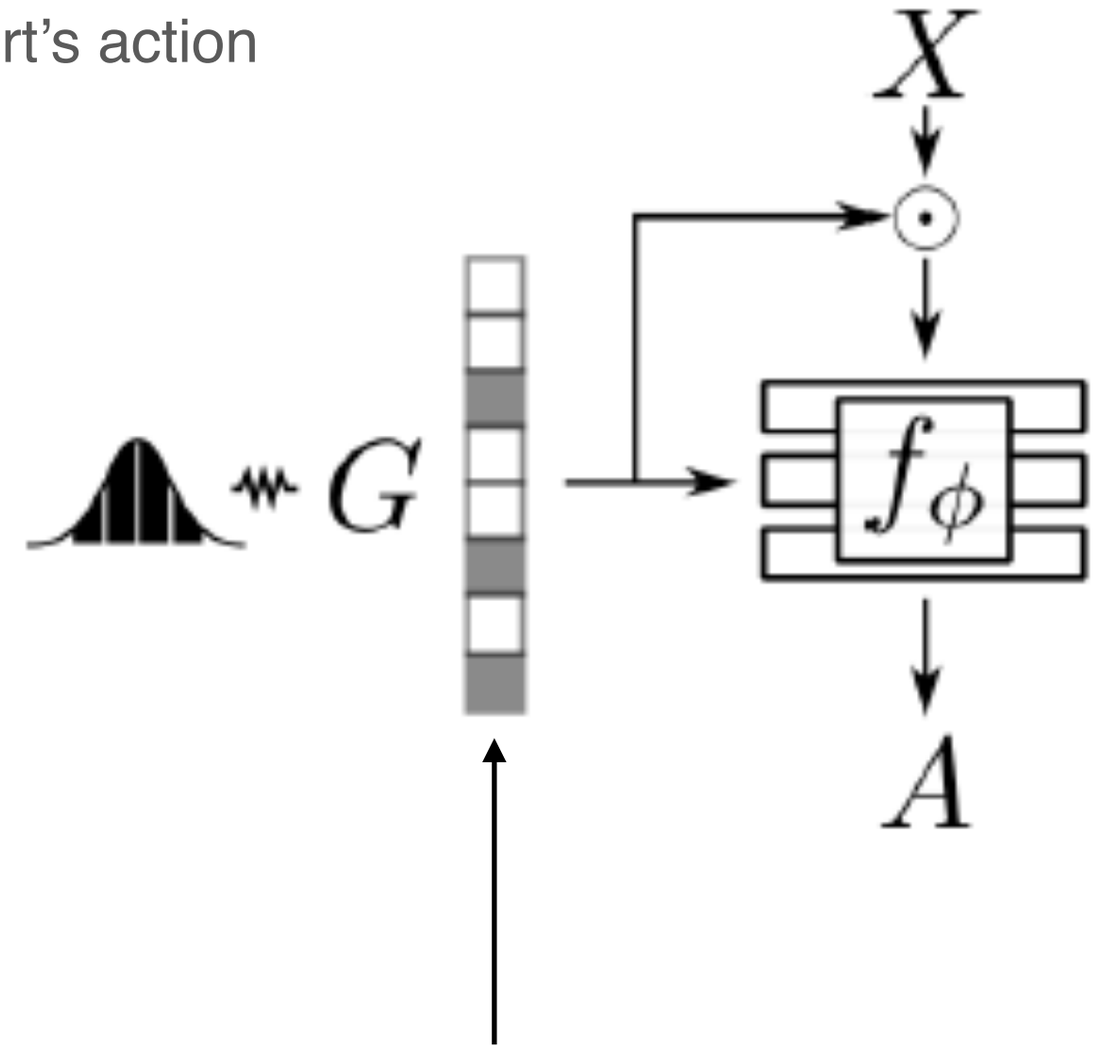
Sampling masking vectors and regressing to actions using the masked state we sampled.

Causal Confusion in Imitation Learning,De Haan et al

Learning a structure-parametrized policy

Input: the mask, and the masked state features

Output: the expert's action



Causal Confusion in Imitation Learning,De Haan et al

Finding the right causal structure

We need to find the right causal graph structure, in other words, the right state feature masking.

We will learn a mapping from graph structure to its likelihood.

We set the graph likelihood to be proportional to exponentiated reward: $p(G) \propto \exp\langle w, G\rangle$

$$p(G) = \prod_i p(G_i) = \prod_i \text{Bernoulli}(G_i|\sigma(w_i/\tau))$$

$$\begin{aligned}
\pi(G) &= \frac{\exp(\langle w, G\rangle + b)/\tau}{\sum_{G'} \exp(\langle w, G'\rangle + b)/\tau} \\
&= \frac{\exp\langle w, G\rangle/\tau}{\sum_{G'} \exp(\langle w, G'\rangle/\tau} \\
&= \frac{\prod_i \exp(w_i G_i/\tau)}{\sum_{G'} \prod_i \exp(w_i G'_i/\tau)} \\
&= \frac{\prod_i \exp(w_i G_i/\tau)}{\prod_i \sum_{G'_i} \exp(w_i G'_i/\tau)} \\
&= \prod_i \frac{\exp(w_i G_i/\tau)}{\sum_{G'_i} \exp(w_i G'_i/\tau)} \\
&= \prod_i \frac{\exp(w_i G_i/\tau)}{1 + \exp w_i/\tau} = \prod_i \text{Bernoulli}(G_i|\sigma(w_i/\tau))
\end{aligned}$$

Causal Confusion in Imitation Learning,De Haan et al

Finding the right causal structure

We need to find the right causal graph structure, in other words, the right state feature masking.

We will learn a mapping from graph structure to its likelihood.

We set the graph likelihood to be proportional to exponentiated reward:

$$p(G) = \prod_i p(G_i) = \prod_i \text{Bernoulli}(G_i | \sigma(w_i/\tau))$$

To obtain task rewards we will be deploying the correpsonding policy, obtaining trajectories and scoring their rewards.

Algorithm 2 Policy execution intervention

Input: policy network f_ϕ s.t. $\pi_G(X) = f_\phi([X \odot G, G])$
Initialize $w = 0, \mathcal{D} = \emptyset$.
for $i = 1 \ldots N$ **do**
  Sample $G \sim p(G) \propto \exp\langle w, G \rangle$.
  Collect episode return R_G by executing π_G.
  $\mathcal{D} \leftarrow \mathcal{D} \cup \{(G, R_G)\}$
  Fit w on $\mathcal{D}$ with linear regression.
end for
Return: $\arg\max_G p(G)$

Causal Confusion in Imitation Learning,De Haan et al