ICML 2018 Notes
Stockholm, Sweden

David Abel[*]
david_abel@brown.edu

July 2018

Contents

[*] http://david-abel.github.io/

This document contains notes I took during the events I managed to make it to at ICML in Stockholm, Sweden. Please feel free to distribute it and shoot me an email at david_abel@brown.edu if you find any typos or other items that need correcting.

....

1 Conference Highlights

Some folks jokingly called it ICRL this year — the RL sessions were in the biggest room and apparently had the most papers. It's pretty wild. A few of my friends in RL were reminiscing over the times when there were a dozen or so RL folks at a given big ML conference. My primary research area is in RL, so I tend to track the RL talks most closely (but I do care deeply about the broader community, too), All that being said, these notes are *heavily* biased toward the RL sessions. Also, I was spending quite a bit more time prepping for my talks/poster sessions so I missed a bit more than usual.

Some takeaways:

- I'd like to see more explanatory papers in RL – that is, instead of focusing on introducing new algorithms that perform better on our benchmarks, reflecting back on the techniques we've introduced and do a deep analysis (either theoretical or experimental) to uncover what, precisely, these methods do.
- I'm going to spend some time thinking about what it would look like to make foundational progress in RL without MDPs at the core of the result, (there's some nice work out there already [30]).
- Lots of tools are sophisticated and robust enough to make a huge impact, now. If you're into AI for the long haul Utopia style vision of the future, now is a good time to start thinking deeply about how to help the world with the tools we've been developing. As a start take a look at the AI for Wildlife Conservation workshop (and comp sust community[1]).
- Sanjeev Arora's Deep Learning Theory tutorial and Ben Recth's Optimization tutorial were both excellent – I'd suggest taking a look at each if you get time. The main ideas for me were (Sanjeev) we might want to think about doing unsupervised learning with more connection to downstream tasks, and (Ben) RL and Control theory have loads in common, and the communities should talk more.

2 Tuesday July 10th

It begins! Tuesday starts with Tutorials (I missed the morning session due to jet lag). I'll be attending the Theory of Deep Learning tutorial and the Optimization for Learning to Control tutorial.

[1] http://www.compsust.net/

2.1 Tutorial: Toward Theoretical Understanding of Deep Learning

Sanjeev Arora is speaking.[2]
Some Terminology:

- Parameters of deep net
- $(x_1, y_1) \ldots, (x_i, y_i)$ iid training from distribution $\mathcal{D}$
- $\ell(\theta, x, y)$: Loss function
- Objective: $\arg\min_\theta E_i \left[\ell(\theta, x_i, y_i)\right]$
- Gradient Descent:

$$\theta^{t+1} \leftarrow \theta_t - \eta \nabla_\theta \mathbb{E}_i \left[\ell(\theta_t, x_i, y_i)\right] \tag{1}$$

Point: Optimization concepts already shape deep learning.

Goal of Theory: Theorems that sort through competing intuitions, lead to new insights and concepts. A mathematical basis for new ideas.

Talk Overview:

1. *Optimization*: When/how can it find decent solutions. Highly nonconvex.
2. *Overparameterization/Generalizations*: When the # parameters >> # training samples. Does it help? Why no nets generalize?
3. *The Role of Depth*
4. *Unsupervised Learning/GANs*
5. *Simpler methods to Replace Deep Learning*

2.1.1 Optimization

Point: Optimization concepts have already helped shape deep learning.
Hurdle: Most optimization problems are non-convex. So, we don't expect to have polynomial time algorithms.

Possible Goals of Optimization:

- Find critical point $\nabla = 0$.
- Find local optimum: ∇^2 is positive semi-definite.
- Find global optimum, θ^*.

Assumptions about initialization:

- Prove convergence from all starting points θ_0

[2] Video will be available here: `https://icml.cc/Conferences/2018/Schedule?type=Tutorial`.

- Prove random initial points will converge.
- Prove initialization from special initial points.

Note: if optimization is in $\mathbb{R}^d$, then you want run time $poly(d, 1/\varepsilon)$, where $\varepsilon =$ accuracy. The naive upper bound is exponential in $\exp d/\varepsilon$.

Curse of Dimensionality: In $\mathbb{R}^d$, $\exists \exp(d)$ directions whose parwise angle is $> 60°$. Thus, $\exists exp(d/\varepsilon)$ special directions s.t. all directions have angle at most ε with one of these (an "ε-cover").

Black box for analysis of Deep Learning. Why: don't know the landscape, really, just the loss function. We have basically no mathematical characterization of (x, y), since y is usually a complicated function of x (think about classifying objects in images: x is an image, y is "dog").

Instead, we can get: $\theta \to f \to f(\theta), \nabla f_\theta$. Using just this blackbox analysis, we can't get global optimums.

Gradient Descent:

- $\nabla \neq 0$: so, there is a descent direction.
- But, if ∇^2 is high, allows ∇ to fluctuate a lot!
- So, to ensure *descent*, we must take small steps determined by smoothness:

$$\nabla^2 f(\theta) \leq \beta I \tag{2}$$

Claim 2.1. *If $\eta = 1/2\beta$, then we can achieve $|\nabla f| < \varepsilon$, in # steps proportional to β/ε^2.*

Proof.

$$\begin{aligned} f(\theta_t) - f(\theta_{t+1}) &\geq \nabla f(\theta_t)(\theta_{t+1} - \theta_t) - \frac{1}{2}\beta|\theta_t - \theta_{t+1}|^2 \\ &= \eta|\nabla_t|^2 - \frac{1}{2}\beta\eta^2|\nabla_t|^2 = \frac{1}{2\beta} = |\nabla_t|^2 \end{aligned}$$ □

But, the solution here is just a critical point, which is a bit too weak. One idea to improve: avoid saddle points, as in Perturbed SGD introduced by Ge et al. [18].

What about 2nd order optimization? Like the Newton Method. So, we instead consider:

$$\theta \to f \to f(\theta), \nabla f_\theta, \nabla^2 f_\theta, \tag{3}$$

which lets us make stronger guarantees about solutions at the expense of additional computation.

Non-black box analyses. Lots of ML problems that are subclasses of depth two neural networks:

- Make assumptions about net's structure, data distribution, etc.
- May use different algorithms from SGD.

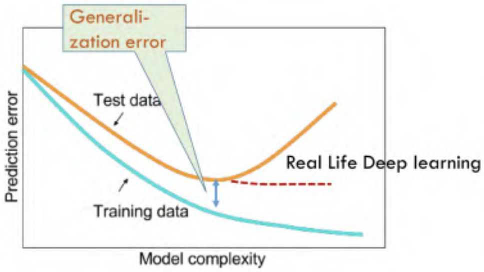


Figure 1: Classical story of overfitting in Machine Learning.

Problem: Matrix completion. Suppose we're given an $n \times n$ matrix M of rank r with some missing entries:

$$M = U \cdot V^T \tag{4}$$

Goal is to predict the missing entries. → subclass of learning depth two linear nets! Feeding $1 - hot$ inputs into unknown net; setting output at one random output node. Then, learn the net! Recent work: all local minima for this problem are global minima, proven by [19] (for an arbitrary starting point).

Theorems for learning multilayer nets? Yes! But usually only for linear nets. Overall net: product of matrix transformation. Some budding theory:

- Connection to physics: natural gradient/Lagrangian methods.
- Adversarial examples and efforts to combat them.
- Optimization for unsupervised learning
- Connections to information theory.

2.1.2 Overparameterization and Generalization Theory

Guiding Q: Why is it a good idea to train VGG19 (20mil parameters) on CIFAR 10?

Overparameterization may help optimization: folklore experiment [31].

1. Generate labeled data by feeding random input vectors into depth 2 net with hidden layer of size n.
2. Difficult to train a new net using this labeled data with same # of hidden nodes.
3. **But**: much easier to train a new net with bigger hidden layer.
4. (Still no theorems to explain this!)

But of course, textbooks warn us: Large models can “overfit”:
But, recent work shows the excess capacity of networks is still there! [47]:
Hope: an explanation of these concepts will give us a better notion of a “well-trained net”.

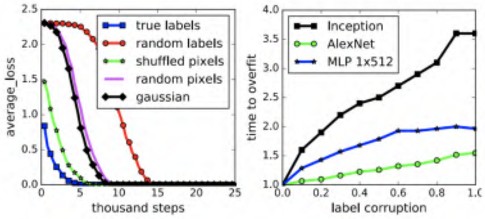


Figure 2: Excess capacity experiment from Zhang et al. [47]

Effective Capacity: roughly, log(# distinct *a priori* models). Generalization theory tells us:

$$\text{Test loss - training loss} \leq \sqrt{\frac{N}{m}}. \tag{5}$$

Where m = # training samples, whereas N = # of parameters, VC Dimension, Rademacher complexity.
Worry, though: for Deep Nets, N dominates so much that this is vacuous. Idea, via proof sketch:

- Fix a network's parameters θ.
- Take i.i.d. samples S of m datapoints:

$$\text{Error}_\theta = \text{avg error on } S. \tag{6}$$

- By concentration bounds, for fixed net θ, we get our usual concentration inequalities:

$$\Pr(d(\theta, \theta^*) \leq \varepsilon) \geq 1 - \exp(-\varepsilon^2 m). \tag{7}$$

- *Complication*: Net depends on training samples S.

- *Solution*: Union bound over all θ.
- Thus, if # possible $\theta = \underbrace{\mathcal{W}}_{capacity}$, suffices to let $m > \mathcal{W}/\varepsilon^2$. But then this is the same for effectively all nets.

Current method of generalization theory: find property Φ that only obtains in a few neural networks, and correlates well with generalization. Then, we can use Φ to compute upper bounds on "very few" networks, and thus lowers effective capacity.

Von Neumann: "Reliable Machines and unreliable components. We have, in human and animal brains, examples of large and relatively reliable systems constructed from individual components, the neurons, which would appear to be anything but reliable... In communication theory this can be done by properly introduced redundancy".

New Idea: Compression-based methods for generalization bounds, introduced at ICML this year in Arora et al. [5]. The bound is roughly:

$$capacity \approx \left(\frac{\text{depth} \times \text{activation contraction}}{\text{layer cushion} \times \text{interlayer cushion}}\right)^2 \tag{8}$$

Concluding thoughts on generalization:

- Recent progress! But final story still to be written.
- We don't know why trained nets are noise stable.
- Quantitative bounds too weak to explain why net with 20mil params generalizes with 50k training dataset.
- Argument needs to involve more properties of training algorithm and/or data distribution.

2.1.3 The Role of Depth in Deep Learning

Ideal Result: exhibit natural learning problems that cannot be done using depth d but can be done with depth $d + k$.
Critically, we're talking about *natural learning problems*, which don't tend to have nice mathematical formalizations. Recent work shows this is true for non-natural cases [16].

Q: Does more depth help or hurt in deep learning?

- Pros: better expressiveness
- Cons: more difficult to optimize
- New Result! Arora et al. [4] show that increasing depth can sometimes accelerate the optimization, including for classical convex problems.

Consider regression, in particular, ℓ_p regression:

$$L(w) = \mathbb{E}_{(x,y)\sim D}\left[\frac{1}{p}(x^T w - y)^p\right] \tag{9}$$

Now, we'll replace this with a depth-2 linear circuit – so, we replace w by $w_1 \cdot w_2$ (overparameterize!):

$$L(w) = \mathbb{E}_{(x,y)\sim D}\left[\frac{1}{p}(x^T w_1 w_2 - y)^p\right] \tag{10}$$

Why do this? Well, the path that gradient descent might take, this might be easier. Gradient descent now amounts to:

$$w_{t+1} = w_t - \underbrace{\rho_t \nabla_{w_t}}_{\text{adaptive learning rate}} - \underbrace{\sum_{\tau=1}^{t-1} \mu^{(t,\tau)} \nabla_{w_t}}_{\text{memory of past gradients}} \tag{11}$$

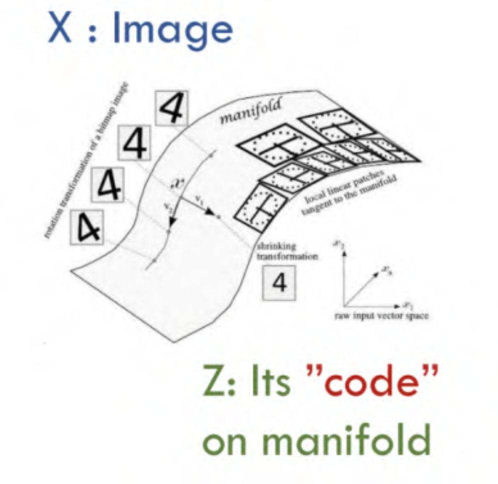

X : Image
rotation transformation of a bitmap image
manifold
local linear patches tangent to the manifold
shrinking transformation
raw input vector space
Z: Its "code" on manifold

Figure 3: Manifold learning

2.1.4 Theory of Generative Models and Adversarial Nets

Unsupervised learning motivation: "manifold assumptions"
Goal: using a large unlabeled dataset, learn the mapping from image to codes. The hope here is that the code is a good downstream substitute for X in classification tasks.

Generative Adversarial Nets (GANs) [20].

- *Motivation*: Avoid likelihood objective, which favors, for instance, outputting fuzzy images.
- Instead of log-likelihood, use power of discriminative learning. item New Objective:

$$\min_{u \in \mathcal{U}} \max_{v \in \mathcal{V}} \mathbb{E}_{x \dim \mathcal{D}_{real}}[D_v(x)] - \mathbb{E}_h[D_v(G_u(h))] \tag{12}$$

Generator "wins" if objective ≈ 0, and further training if discriminator doesn't help (reached equilibrium).

Q: What spoils a GAN trainers day? A: **Mode collapse**! Idea: since discriminator only learns from a few samples, it may be unable to teach generator to produce distribution $\mathcal{D}_{synth}$ with sufficiently large diversity.

New insights from theory: the problem is *not* with # training samples but size/capacity of the discriminator!

Theorem 2.2. *Arora et al. [3] If discriminator size = N, then $\exists$ a generator that generates a distribution supported on $O(NlogN)$ inputs and still wings against all possible discriminators.*

Main Idea: small discriminators inherently incapable of detecting mode collapse. Theory suggests GANs training objective not guaranteed to avoid mode-collapse. But, does this actually happen?

A: Yep! Recall the Birthday paradox. If you put ≥ 23 people in a room, the chance is > 0.5 that two of them share a birthday. Note that $23 \approx \sqrt{365}$.

Thus: if a distribution is supported on N images. Then Pr(sample of size $\sqrt{N}$ has a duplicate image $\geq 1/2$.

Briefly: New Story needed for Unsupervised Learning

Unsupervised Learning motivation: the "manifold" assumption.

Possible hole: For the code learned to be good, then $p(X, Z)$ needs to be learnt to *very high numerical accuracy*, since you're going to use the code in a down stream task. But this doesn't really happen!

So: the usual story is a bit off.

Food for thought:

- Maximizing Log Likelihood may lead to little unstable insight into the data.
- How do we define the utility of GANs?
- Need to define unsupervised learning using a "utility" approach.

2.1.5 Deep Learning Free

Consider two sentence that people find similar:

A lion rules the jungle.
The tiger hunts in the forest.

Problem: no words in common. So, how should we capture text/sentence similarity?

Usual story: Text embedding.

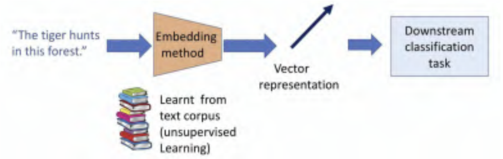

"The tiger hunts in this forest."
Embedding method
Vector representation
Downstream classification task
Learnt from text corpus (unsupervised Learning)

Figure 4: Text Embedding

(Ben Recht?) Linearization Principle: "Before committing to deep model figure out what the linear methods can do." But, Sanjeev says, Ben doesn't actually say this is his philosophy.

The point of learning a representation is that the true structure of the data emerges, and classification becomes easy. But: the downstream task isn't always known ahead of time! So, maybe the representation should capture all or most of the information (like a bag of words).

Recovery algorithm:

$$\min |x|_1 \text{ s.t. } Ax = b \tag{13}$$

But, Calderbank et al. [12] showed that linear classification on compressed vector Ax is as good as x.

Connection to RL: simple linear models in RL can beat the state of the art Deep RL on some simple tasks. Linearization principle, applied! See Ben's talk (next section).

2.1.6 Conclusion

What to work on:

1. Use Physics/PDE insights such as calculus of variations (Lagrangians, Hamiltonians)
2. Look at unsupervised learning! Yes everything is NP-Hard and new but that's how we'll grow.
3. Theory for Deep Reinforcement Learning. Currently very little!
4. Going beyond (3.), design interesting models for interactive learning of language/skills. Both theory and applied work are missing some basic ideas.
5. "Best theory will emerge from engaging with real data and real deep net training. (Nonconvexity and attendant complexity seems to make armchair theory less fruitful.)"
6. Hilbert: "In mathematics there is no 'ignorabiums'"

....

2.2 Tutorial: Optimization Perspectives on Learning to Control

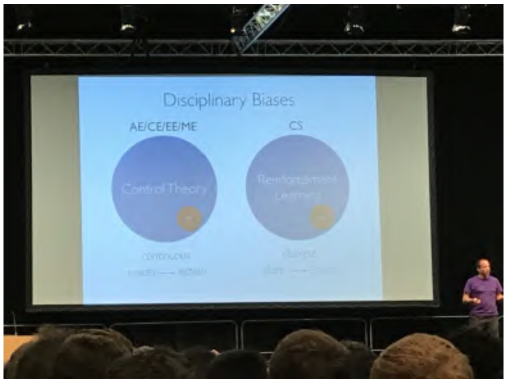
The speaker is Benjamin Recht.

2.2.1 Introduction: RL, Optimzation, and Control

Preface of the talk: if you grew up in continuous control, what would you want to/need to know about Reinforcement Learning?

The games we've been successful on (Atari, Go, Chess, etc.) are too structured – what happens when we move out of games and into the real world? In particular, move into settings where these systems interact with people in a way that actually a major impact on the lives of lots of people.

Definition 1 (Reinforcement learning): *RL (or control theory?) is the study of how to use past data to enhance the future manipulation of a dynamical system?*



Disciplinary Biases

AE/CE/EE/ME

Control Theory

continuous

model → action

CS

Reinforcement Learning

discrete

data → [illegible]

Figure 5: RL vs. Control

If you come from a department with an "E" in it, then you study CT, and RL is a subset. If you come from a CS department, then you study RL, and CT is a subset.
Today's Talk: Try to unify these camps and point out how to merge their perspectives.

Main research challenge: what are the fundamental limits of learning systems that interact with the environment?

> **Definition 2** (Control Theory): *The study of dynamical systems with inputs.*
>
> *Example:* $x_{t+1} = f(x_t, u)$.

> **Definition 3** (Reinforcement Learning): *The study of discrete dynamical systems with inputs, where the system is described as a Markov Decision Process (MDP).*
>
> *Example:* $s_{t+1} = p(s_{t+1} \mid s_t, a)$.

Main difference: are we discrete or continuous?

Optimal Control:

$$
\begin{aligned}
\min \mathbb{E}_e \left[\sum_{t=1}^{T} C_t(x_t, u_t) \right], \\
\text{s.t. } x_{t+1} = f_t(x_t, u_t, e_t) \\
\text{s.t. } u_t = \pi_t(\tau_t).
\end{aligned}
$$

Where:

- So, C_t is the cost. If you maximize it, it's called reward.
- e_t is a noise process

- f_t is the state transition function
- $\tau_t = (u_1, u_2, \ldots, u_t)$ is a trajectory
- $\pi_t(\tau_t)$ is the policy

Example: Newton's Laws define our model. So:

$$\begin{aligned} z_{t+1} &= z_t + v_t \\ v_{t+1} &= v_t + o_t \\ mo_t &= u_t \end{aligned}$$

With cost defined by reaching a particular location:

$$\text{minimize} \sum_{t=0}^{T} x_t^2 + ru_t^2 \tag{14}$$

Subject to some simple constraints (time, energy, etc.). The cost function isn't given, typically–we assume that it requires care in designing.

The example we just introduced is the "Linear-Quadratic Regulator":

Definition 4 (Linear Quadratic Regulator (LQR)): *Minimize a quadratic cost subject to linear dynamics. In some sense, the canonical, simple problem (similar to grid world in RL?)*

Generic solutions with known dynamics:

1. Batch Optimization
2. Dynamic Programming

Remember, f is the state transition function ($\mathcal{T}$ from MDPs).

Major Challenge: How to we perform optimal control when the system is unknown? (When f is unknown?)

So, now: recreate RL to solve this challenge.

Example: Consider the success story of cooling off data centers – here, the dynamics are unknown. How could we solve this?

- Identify everything: PDE Control, High performance dynamics.
- Identify a coarse model: model predictive control.
- We don't need no stinking model: RL, PID control.

  PID control works: 95% of all industrial control applications are PID controllers.

  Some Qs: How much needs to be modeled for more advanced control? Can we learn to compensate for poor models or changing conditions?

Learning to control problem:

$$\begin{aligned}\mathbb{E}_e\left[\sum_{t=1}^{T} C_t(x_t, u_t)\right]& \\ \text{s.t. } x_{t+1} &= f_t(x_t, u_t, e_t) \\ \text{s.t. } u_t &= \pi_t(\tau_t).\end{aligned}$$

Oracle: you can generate N trajectories of length T.

Challenge: Build a controller with smallest error with fixed sampling budget ($N \times T$). So, what is the optimal estimation/design scheme?
Big Question: How many samples are needed to solve the above challenge?

Definition 5 (The Linearization Principle): *"If a machine learning algorithm does cray thing when restricted to linear models, it's going to do crazy things on complex nonlinear models, too."*

Basically: would you believe someone had a soot SAT solver if it can't solve 2SAT problems?

2.2.2 Different Approaches to Learning to Control

Recall again the LQR example. Three general things that might work:

1. *Model-based*: Fit model from data.
2. *Model-free*:
   (a) Approximate Dynamic Programming: Estimate cost from data.
   (b) Direct Policy Search: search for actions from data.

Model-Based RL:

- Idea: Collect some simulation data, should have $x_{t+1} \approx \phi(x_t, u_t) + v_t$).
- One idea is to fit dynamics with supervised learning:

$$\hat{\phi} = \arg\min_{\phi} \sum_{t=0}^{N} |x_{t+1} - \phi(x_t, u_t)|^2 \tag{15}$$

- Then, solve approximate problem, same as LQR but use $\hat{\phi}$ as the model.

Dynamic Programming:
Let's first suppose everything is known, and just consider the DP problem. Then, we can define our usual Q function as this expected cost:

$$Q_1(x, u) = \mathbb{E}_e\left[\sum_{t=1}^{T} C_t(x_t, u_t)\right] \tag{16}$$

If we continue this process, we end up with the true recursive formulation of Q values:

$$Q_t(x,u) = \mathbb{E}_e C_t(x_t,u_t) + \min_{u'} \left[\sum_{t=1}^{T} Q_{t+1}(f_t(x,u,e),u') \right]. \tag{17}$$

Optimal policy, then:

$$\pi_k(\tau_k) = \arg\min_u Q_t(s_t,u). \tag{18}$$

People love LQR! Again suppose final cost is actually quadratic:

$$\min \mathbb{E} \left[\sum_{t=1}^{T} x_t Q x_t + u_t R u_t + x_T P_t x_T \right] \tag{19}$$

Well, quadratics are closed under minimization, so:

$$Q_t(x,u) = C_t(x,u) + \min_{u'} \mathbb{E}_t \left[Q_{t+1}(f_t(x,u,e),u') \right]. \tag{20}$$

Because quadratics are well behaved, we get a closed form for the optimal action. A couple nice things:

- DP has simple form because quadratics are miraculous
- Solution is independent of noise variance
- For finite time horizion we could solve this with a variety of batch solvers
- Note that the solution is only relative to one time horizon.

Approximate Dynamic Programming

Bellmean Equation:

$$Q(x,u) = C(x,u) + \gamma \mathbb{E}_e \left[\min_{u'} Q(f(x,u,e),u') \right] \tag{21}$$

Optimal Policy:

$$\pi(x) = \arg\min_u Q(x,u) \tag{22}$$

Applying gradient descent yields Q-learning.

Direct Policy Search

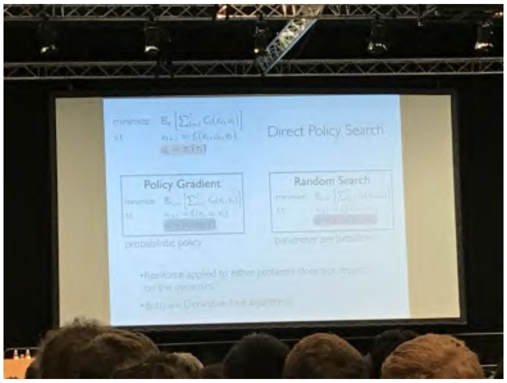
The final idea: just search for good policies correctly. Basically: sampling to search.

New Problem:

$$\min_{z \in \mathbb{R}^d} \Phi(z). \tag{23}$$

Note that this problem is equivalent to optimizing over probability distributions:

$$\min_{p(z)} \mathbb{E}\left[\Phi(z)\right] \tag{24}$$



minimize $\mathbb{E}_e\left[\sum_{t=1}^{T} C_t(x_t,u_t)\right]$

s.t. $x_{t+1} = f_t(x_t,u_t,e_t)$

$u_t = \pi_t(\tau_t)$

Direct Policy Search

Policy Gradient

minimize $\mathbb{E}_{e,u}\left[\sum_{t=1}^{T} C_t(x_t,u_t)\right]$

s.t. $x_{t+1} = f_t(x_t,u_t,e_t)$

[illegible]

probabilistic policy

Random Search

minimize [illegible]

s.t. [illegible]

[illegible]

parameter perturbation

- Reinforce applied to either problems does not depend on the dynamics
- Both are Derivative-free algorithms

Figure 6: Different approaches to direct finding a policy.

Notably, though, the above is bounded by the conditional form:

$$\min_{p(z)} \mathbb{E}\left[\Phi(z)\right] \leq \min_{\theta} \mathbb{E}_{p(z;\theta)}\left[\Phi(z)\right] = f(\theta). \tag{25}$$

Then, we can use function approximators that might not capture optimal distribution. Can build stochastic gradient estimates by sampling:

$$\nabla f(\theta) = \mathbb{E}_{p(z;\theta)}\left[\Phi(z)\nabla_\theta \log(p(z;\theta))\right]. \tag{26}$$

Thus, we introduce REINFORCE [44]:

1. Sample: $z_T \sim p(z;\theta_k)$
2. Compute: $G(z_k;\theta_k) = \Phi(z-k)\nabla_\theta \log p(z_k;\theta_k)$
3. Update: $\theta_{k+1} = \theta_k - \alpha G(z_k,\theta_k)$.

REINFORCE is used at the heart of both policy gradient algorithms and random search algorithms. To do policy gradient, we replace our deterministic policy with a stochastic one:

$$u_t \sim p(u \mid x_t;\theta). \tag{27}$$

Conversely, if we perturb the *parameters* of the policy, we get random search.

REINFORCE is not magic:

- What is the variance of the estimator of the gradient?
- What is the approximation error?
- Necessarily becomes derivative free as your accessing the decision variable by sample.
- But! It's certainly super easy.

2.2.3 Learning Theory

What can we say about sample complexity? In particular, what can we say about the sample complexity of each of the three classes of methods we introduced in the previous section? (Approximate DP, Model-based, and Policy Search).

One idea to estimate sample complexity: do parameter counting. Shown in Table 1.

Algorithm Class	Samples per iteration	Parameters	"optimal error after T"
Model-based	1	d^2p	$\sqrt{\frac{d^2p}{T}}$
ADP	1	dp	$\sqrt{\frac{dp}{T}}$
Policy search	1	dp	$\sqrt{\frac{dp}{T}}$

Table 1: *Discrete case:* Approximate Sample Complexity of algorithms based on parameters.

Where the above "error" column, is a super rough approximation based on # parameters alone.

What about when we move to the *continuous* case? Shown in Table **??**.

Algorithm Class	Samples per iteration	LQR Parameters	"optimal error after T"
Model-based	1	$d^2 + dp$	$C\sqrt{\frac{d+p}{T}}$
ADP	1	$\binom{d+p}{2}$	$C(d+p)/\sqrt{T}$
Policy search	1	dp	$C\sqrt{dp/T}$

Table 2: *Continuous case:* Approximate Sample Complexity of algorithms based on parameters.

Let's return to LQR and think about sample complexity. Ben ran some experiments on a double integrator task from each of the three algorithm classes, and after about 10 samples, ADP and model-based solved the problem, whereas Policy Gradient did very poorly.

Lance Armstrong: "Extraordinary Claims Require Extraordinary Evidence" ("only if you prior is correct!" – Ben).

OpenAI quote on trickiness of implementing RL algorithms: "RL results are tricky to reproduce performance is very noisy, algorithms have many moving parts which allow for subtle bugs, and many papers don't report all the required tricks." Also see Joelle Pineau's keynote talk at ICLR.

Ben's Q: Is there a better way? Can we avoid these pitfalls? A: Yes! Let's use models.

2.2.4 Model-Based RL To The Rescue

Recall, the main idea:

1. Idea: Collect some simulation data, should have $x_{t+1} \approx \phi(x_t, u_t) + v_t$).

2. One idea is to fit dynamics with supervised learning:

$$\hat{\phi} = \arg\min_{\phi} \sum_{t=0}^{N} |x_{t+1} - \phi(x_t, u_t)|^2 \tag{28}$$

3. Then, solve approximate problem, same as LQR but use $\hat{\phi}$ as the model.

The hard part here is what control problem do we solve? We know our model isn't perfect. Thus we need something like Robust Control/Coarse-ID control.

In Coarse-ID control:

- solve $\min_u x^* Q x$ subject to $x = Bu + x_0$, with B unknown.
- Then, collect data: $D = \{(x_1, u_1) \dots (x_i, u_i)\}$.
- Estimate B:
$$\hat{B} = \min_B \sum_{i=1}^{n} ||Bu)i + x_0 - x_i||^2 \tag{29}$$
- Guarantee $||B - \hat{B}|| \leq \varepsilon$ w/ $\Pr 1 - \delta$.

Then, we can translate this into a robust optimization problem:

$$\min_u \sup_{||\Delta_B \leq \varepsilon} ||\sqrt{Q}(x - \Delta_B u)||, \tag{30}$$

subject to $x = \hat{B}u + x_0$. We can then relax this via the triangle inequality intto a convex problem:

$$||\sqrt{Q}x|| + \varepsilon\lambda||u||, \tag{31}$$

subject to the same constraint. They show how you can translate estimation error into control error in LQR systems – sort of like the simulation lemma from [11]. Yields robust model based control: shows some experimental results, consistently does quite well (definitely better than model-free).

A return to the Linearization Principle: now, what happens when we remove linearity? (QR?). They tried running a random search algorithm on MuJocoo, and found it does better (or at least as good as) natural graident methods and TRPO.

Bens' Proposed Way Forward: Use models. In particular, model-predictive control (MPC):

$$Q_t(x, u) = \sum_{t=1}^{H} C_t(x, u) + \mathbb{E}\left[\min_{u'} Q_{H+1}(f_H(x, u, e)u')\right]. \tag{32}$$

Idea: plan on short time horizon, get feedback, replan.

Conclusions and things left to do:

- Are the coarse-ID results optimal? Even w.r.t. the problem parameters?

- Can we get tight and lower sample complexities for various control problems?
- Adaptive and iterative learning control
- Nonlinear models, constraints, and improper learning.
- Safe explorating, learning about uncertain environments.

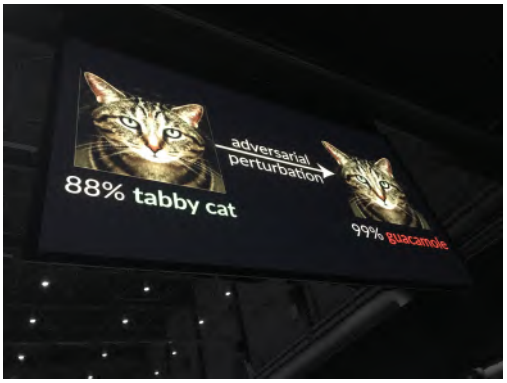
So, lots of exciting things to do! And it's not just RL and not just control theory. Maybe we need a new name that's more inclusive, like "Actionable Intelligence". So, to conclude:

Definition 6 (Actionable Intelligence): *Actionable Intelligence is the study of how to use past data to enhance the future manipulation of a dynamical system.*

Actionable Intelligence interfaces with people, and needs to be trustable, scalable, and predictable.

And that's all for the day.

....



88% tabby cat

adversarial perturbation

99% guacamole

Figure 7: A canonical adversarial example: guacomole cat!

3 Wednesday July 11th

Today the official conference begins! The morning session included the opening remarks and a keynote talk on AI & Security by Dawn Song — I'm still recovering from jetlag so I sadly missed these, but if I get a chance I'll watch the videos and add some notes.

3.1 Best Paper 1: Obfuscated Gradients Give a False Sense of Security [7]

The speaker is Nicholas Carlini, joint with Anish Athalye and David Wagner.

Focus: adversarial examples.

Q: Why should we care about adversarial examples?

1. A1: Make ML robust!
2. A2: Make ML better! (Even ignoring security, we should still want ML to not make these mistakes)

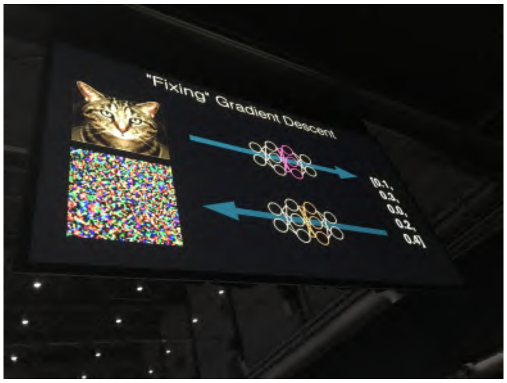
In light of the presence of these examples, prior work have looked into defenses against examples. At ICLR this year, there were 13 defense papers: 9 white box (no theory). In this talk: we show they're broken.

This talk: *How* did we evade these defenses? *Why* were we able to evade them?

How: 7/9 of the ICLR defenses uses obfuscated gradients.

Definition 7 (Obfuscated Gradients): *There are clear global gradients, but local gradients are highly random and directionless.*

So, new attack: "fixing" gradient descent. Idea: run the image through the network to obtain a probability distribution. Then, run it backward through a *new* network, almost identical to the original, but with obfuscated gradients. Using this we can still generate an adversarial image:



"Fixing" Gradient Descent

Figure 8: Using obfuscated gradients to generate adversarial examples: the orange layers are new layers, replaced with obfuscated gradients.

Why: what can we learn? What went wrong in the prior papers?

The usual test:

```
acc, loss = model.eval(x_test, y_test)
```

won't cut it anymore! The only thing that matters is robustness against an adversary *targeting the defense.*

Instead: the purpose of a defense evaluation is to **fail** to show the defense is **wrong**.

Q: What metric should we optimize?

A: **Threat Model**:

> **Definition 8** (Threat Model): *A specific set of assumptions we place on an adversary.*

In the context of adversarial examples, we should be mentioning:

- Perturbation bounds.
- Model access and knowledge.

The threat model must assume the attacker has read the paper and knows the defender is using those techniques to defend.

Conclusion

1. A paper can only do so much evaluation.
2. We need more re-evaluation papers! Less new attacks.
3. "Anyone from the most clueless amateur to the best cryptographer can create an algorithm that he[/she] can't break" – Bruce Schneier
4. One Challenging Suggestion of a defense to break: Defense-GAN on MNIST [37].

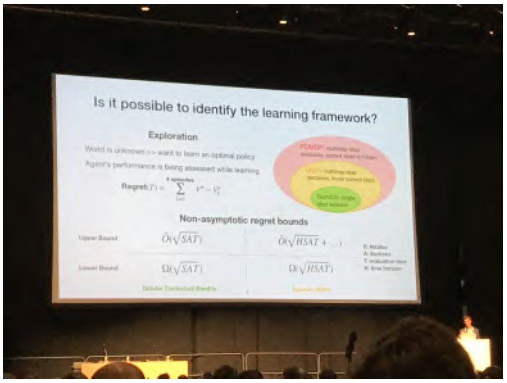
3.2 Reinforcement Learning 1

Now for the first RL session (of many!). RL is in the biggest room this year!

3.2.1 Problem Dependent RL Bounds to Identify Bandit Structure in MDPs [46]

The speaker is Andrea Zanette, joint work with Emma Brunskill.

The main idea: can we design algorithms that inherit better performance on easy MDPs?



Is it possible to identify the learning framework?

Exploration

World is unknown => want to learn an optimal policy

Agent's performance is being assessed while learning

$$\text{Regret}(T) = \sum_{i=1}^{\#\text{ episodes}} V^* - V_i^{\pi}$$

POMDP: multi-step decisions, current state is hidden
MDP: multi-step decisions, know current state
Bandit: single-step decision

Non-asymptotic regret bounds

Upper Bound	$\tilde{O}(\sqrt{SAT})$	$\tilde{O}(\sqrt{HSAT} + \ldots)$
Lower Bound	$\Omega(\sqrt{SAT})$	$\Omega(\sqrt{HSAT})$
	Tabular Contextual Bandits	Episodic MDPs

S: #states
A: #actions
T: evaluation time
H: time horizon

Figure 9: Regret bounds for various learning problems.

Definition 9 (Bandit-MDP): *A bandit which we model as an MDP, where $p(s' \mid s, a) = \mu(s')$. But:*

- *The agent doesn't know this: the agent sees episodic MDP, still optimizing for a policy for an H-step horizon.*
- *Agent is unaware it cannot influence the system dynamics.*

Optimism for exploration: optimistic value function = empirical estimate + exploration bonus. Want a smaller exploration bonus, which yields lower optimism.

Goal Construct the smallest exploration bonus that is as tight as possible for the true underlying problem.

One Idea: $\tilde{Q}(s,a) \approx \hat{Q}(s,a) + \frac{H}{\sqrt{n}}$. Other ideas include range(V^*).

Their Solution: $\frac{\text{range}(V^*)+\Delta}{\sqrt{n}}$. The reason they can get away with this is that a Bandit-MDP is not a "worst-case" MDP. In particular:

1. Mistakes in the Bandit-MDP are not very costly. Basically, agent can recover easily after one mistake since the next state is unaffected.
2. Bandit-MDP is a highly mising MDP ($\tilde{V} \to V^*$).

Main result: shrink Δ based on the structure of the underlying problem.

In conclusion:

- Bandits are not identified by a statistical test.
- Bandit structure is identified during learning to speed up the learning itself.

....

3.2.2 Learning with Abandonment [38]

The speaker is Sven Schmit, joint with Ramesh Johari.

The setting: a platform interacts with a user to learn a user's preferences over time. But! There's a risk: if the system upsets the user, then the user will leave the platform.

Some applications: Newsletters, smart energy meters, notifications.

Definition 10 (Single Treshold Model): *User has a threshold $\theta \sim F$, for F unknown. Then:*

- *Platform selections actions, $x_1, x_2, \ldots$.*
- *If $x_t < \theta$, platform receives reward $R_i(x_t)$*
- *Else, process stops:*

$$\arg\max_{x_t} \mathbb{E}\left[\sum_{t=1}^{T} R_i(x_i)\right] \tag{33}$$

They prove which policies are optimal and near-optimal under a variety of variations of the setting.

Q: What if we learn across users?

New Setting:

- Users arrive sequentially, consider constant policy per user.
- Assumptions: $\text{supp}(F) = [0, 1]$
- $p(x) = r(x)(1 - F(x))$ is concave
- Consider regret:

$$\text{regret}(n) = np(x^*) - \sum_{u=1}^{n} p(x_u) \tag{34}$$

Method: discretize the action space and run UCB, KL-UCB, achieve regret bounds inherited from UCB.

....

3.2.3 Lipschitz Continuity in Model-Based RL [6]

The speaker is Kavosh Asadi, joint with Dipendra Misra and Michael Littman.

Focus: model-based approach to RL. That is, we're going to estimate:

$$\hat{T}(s' \mid s, a) \approx T(s' \mid s, a)$$
$$\hat{R}(s, a) \approx R(s, a)$$

With innacurate models, we get two kinds of error:

1. Innaccurate models
2. Innacurate prediction of previous state

The combination of these errors is deadly to model-based RL! And, critically, we'll never have a perfect model.

Main takeaway: Lipschitz-continuity plays a major role in both overcoming the compounding error problem, but also more generally in model-based RL.

Theorem 3.1. *Given a Δ accurate model under the Wasserstein metric:*

$$W(T(\cdot \mid s, a), \hat{T}(\cdot \mid s, a)) \leq \Delta. \tag{35}$$

Also assume we have an approx. Lipschitz model $K(\hat{T})$ and a true Lipschitz model. Then, the error will be:

$$W(T^n(\cdot \mid s, a); \hat{T}^n(\cdot \mid s, a) \leq \Delta \sum_{i=0}^{n-1} K^i \tag{36}$$

Also introduced results about controlling the Lipschitz constant in neural nets, and regarding the Lipschitz nature of the value function and models.

Q (from Rich Sutton): Is this limited to the tabular representation?

A: Our theory works for non-tabular cases and can be applied to models of arbitrary complexity.

....

3.2.4 Implicit Quantile Networks for Distributional RL [13]

The speaker is Will Dabney, joint with George Ostroviski, David Silver and Remi Munos.

Building on DQN: same basic architecture/learning setup. Here they introduce the Implicit Quantile Network (IQN), that builds on C51 and QR-DQN by trying to relax the assumptions made about discretizing the return output distribution.

Main Story: Move from DQN to IQN— make a slight change to the network by going from the mean (DQN) to samples from that return distribution (IQN), using these samples, you solve the quantile regression problem.

Q: How much data do you need? A: Well, the more you take, the better you do. If you increase number of samples *early* you do quite a bit better – later in the learning problem, you don't get much better by adding samples.

Results: They run it on the usual Atari benchmarks, and they bind it halves the gap between DQN and Rainbow.

....

3.2.5 More Robust Doubly Robust Off-policy Evaluation [17]

The speaker is Mohammad Ghavamzadeh, joint with Mehrdad and Yinlam Chow.

Main problem: off-policy evaluation:

- $\zeta = (x_0, a_0, r_0, \ldots)$ is a T step trajectory.
- $R_{0:T-1}(\zeta)$ the return of the trajectory.
- ρ_T^π : the performance of π.
- If $T = O(1/(1-\gamma))$, then ρ_T^π is a good approximation of ρ_∞^π.

Definition 11 (Off-Policy Evaluation): *The problem is to evaluate a policy π_e given a behavior policy π_b.*

The goal, then, is to compute a good estimate of ρ^{π_e} given a set of T step trajectories from data generated by π_b.

Usually for this problem consider an estimator $\hat{\rho}^{\pi_e}$, typically do the MLE.

One method: do importance sampling on the data collected by the behavior policy in updating ρ^{π_e}.

They introduce the More Robust Doubly Robust Estimator: an estimator for both contextual bandits and RL. Prove new bounds and run experiments comparing their estimator to existing estimators.

....

3.3 Reinforcement Learning 2

Next up, more RL (surprise!).

3.3.1 Coordinating Exploration in Concurrent RL [15]

The speaker is Maria Dimakopoulou, joint with Ben Van Roy.

Idea: focus on *concurrent learning* – a case where lots of agents can be run in parallel simultaneously.

Definition 12 (Concurrent RL): *We have:*

- *K agents interacting with different instances of an MDP M.*
- *Multiple concurrent and asynchronous interactions.*
- *Agents are uncertain about P and R about which they share priors.*

If we just run ϵ-greedy across all the concurrent agents, we don't see much of a benefit in exploration due to the lack of coordination.

Main Question: how can we coordinate across the agents?

Introduce: SeedSampling, an algorithm that coordinates exploration across agents.

Three properties needed to coordinate exploration:

1. *Adaptivity*: each agent needs to adapt appropriately to data.
2. *Commitment*: main the intent to carry out action sequences that span multiple periods.
3. *Diversity*: Divide-and-conquer learning opportunities among agents.

SeedSampling: extends PSRL by satisfying each of the above three properties. Each agent starts by sampling a unique random seed. This seed maps to an MDP, thereby diversifying exploratory efforts among agents.

3.3.2 Gated Path Planning Networks [29]

The speaker is Lisa Lee, joint with Emilio Parisotoo, Devendra Chaplot, Eric Zing, and Ruslan Salakhutdinov.

Path Planning: find the shortest set of actions to reach a goal location from a starting state.

Other approaches: (1) A^*, but not differentiable, and (2) Value Iteration Networks (VIN) → differentiable. So, VINs are becoming widespread.

Problem: VINs are difficult to optimize. So, goal here is to make them easier to optimize. In particular, Non-gated RNNs are known to be difficult to optimize.

Proposal: replace non-gated RNN with gated-RNNs, and allow for a larger kernel size, yielding Gated Path-Planning Networks.

Experiments: run in a maze like environment and 3D VizDoom, comparing to VINs, showing consistent improvement. Really thorough experimental analysis, studying generalization, random seed initialization, and stability.

3.4 Deep Learning

Switching up for a bit to the Deep Learning session.

3.4.1 PredRNN++: Towards a Resolution of the Deep-in-Time Dilemma [42]

The speaker is Yunba Wang, joint with Zhifeng Gao, Mingsheng Long, Jianmin Wang and Philip S. YU.

Definition 13 (Spatiotemporal Predictive Learning): *Receive sequence of data, $X_1 \dots X_t$ and predict the next k of the sequence:*

$$\hat{X}_{t+1} \dots \hat{X}_{t+k} = \arg\max_{X_t \dots X_{t+k}} p(X_t \dots X_{t+k} \dots \mid X_1, \dots X_t) \tag{37}$$

Prior architectures: RNNs (Seq2Seq, Convolutional LSTMS), CNNs (adversarial 2D CNNs, 3D CNNs), Others (PredRnn, Video Pixel).

Main problem: Previous model (PredRNN) uses zigzag memory flow. But: for short term → deeper-in-time networks, the gradient vanishes yielding bad *long-term* modeling capability.

Main contribution: Causal LSTM, uses a longer path for short term dynamics.

They run experiments in the Moving MNIST dataset and the KTC action dataset finding consistent significant improvement over relevant baselines.

....

3.4.2 Hierarchical Long-term Video Prediction without Supervision [43]

The speaker is Nevan Whichers, joint work with Ruben Villegas, Dumitru Erhan, and Honglak Lee.

Task: given the first n frames of a video predict the next k frames.

Major problem with prior work: fail to predict far into the future.

They introduce an architecture that encodes current frames and determines loss based on both prediction of the encoding and the original frame (as far as I can tell – the architecture was relatively complex). Also an adversarial component in the mix – I think used to encourage future frames to be indistinguishable to a discriminator.

Experiments: (1) A shape video prediction problem, where there's does extremely well, (2) Human pose prediction dataset, (3) Human Video prediction.

....

3.4.3 Evolving Convolutional Autoencoders for Image Restoration [40]

The speaker is Masanori Suganuma, joint work with Mete Ozay, and Takayuki Okatani.

Goal: Restore an image using deep neural networks.

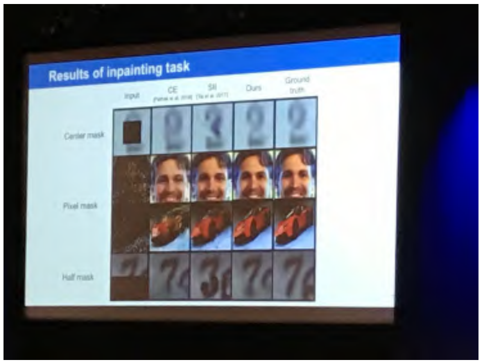
Q: Are standard network architectures optimized enough?

A: Maybe not! Let's do some exploration of possible architecture space.

This work: shows that using an evolutionary approach can evolve useful architectures for the purpose of Convolutional Autoencoders, applied to image restoration.

Idea: Represent a CAE architecture as a Directed Acylcic Graph (phenotype), encoded by a genotype. Then, optimize a genotype using typical evolutionary algorithms.

Experiments: Inpainting, denoising task.



Results of inpainting task

Figure 10: Different image restoration results in the impainting task.

3.4.4 Model-Level Dual Learning [45]

The speaker is Tao Qin, joint with Yingce Xia, Xu Tan, Fei Tian, Nenghai Yu, and Tie-Yan Liu.

Symmetry is beautiful! See: yin-yang, butterfly.

Also, symmetry is useful in AI (primal task → dual task).

Definition 14 (Dual-learning): *A learning framework that leverages the symmetric (primal-dual) structure of AI tasks to obtain effective feedback or regularization signals to enhance learning.*

This work: model-level duality. Duality exists not only in the data, but also at the level of the model. For instance: neural machine translation.

Thus: because of this model level symmetry, we can share knowledge across models. Seems cool! They evaluated one instance of it in a few different experiments, including machine translation, sentiment analysis, and an asymmetric setting (closer to traditional classification).

....

Dave: Taking a short break.

3.5 Reinforcement Learning 3

Alright, a final collection of RL talks for the day.

3.5.1 Machine Theory of Mind [34]

The speaker is Neil Rabinowitz, joint with Frank Perbert, Francis Song, Chiyuan Zhang, Ali Eslami, and Matthew Botvinick.

Preamble: We often find ourselves thinking–"Wow, what is my agent *doing*?"

Q: How should we look at RL agents and diagnose what they do?

A: Well, surely we do this with people all the time. We assign meaning to others' actions regularly. This is commonly called the "Theory of Mind" (by Cog. Pscyhology folks?).

Definition 15 (Theory of Mind): *From Dennett:*

> *Here is how it works: first you decide to treat the object whose behavior is to be predicted as a rational agent; then you figure out what beliefs that agent ought to have, given its place in the world and its purpose. Then you figure out what desires it ought to have, on the same considerations, and finally you predict that this rational agent will act to further its goals in the light of its beliefs. A little practical reasoning from the chosen set of beliefs and desires will in most instances yield a decision about what the agent ought to do; that is what you predict the agent will do.*

> –*Daniel Dennett The Intentional Stance.*

Two camps: (1) Theory of mind, (2) Theory theory (also called “simulation theory”.

Theories of differing complexity: (simple) model-free, (middle) teleological stance, intentional stance, and (complex) recursive stance.

Lots of work in modeling other agents in ML: imitation learning, inverse RL, opponent modeling, multi-agent RL, and more.

This work: taking inspiration from human Theory of Mind – we learn how humans work during our development. We build this strong prior over how to understand other agents.

Desiderata:

1. A system that learns autonomously how to model new agents online
2. Does not simply assume others are noisy-rational utility maximizers with perfect planning
3. Target: from past behaviour, predict future behavior.
4. Goal: build structure that learns a prior which captures general properties of population.
5. Infers a posterior that captures properties of a single agent.

Sally-Anne test [8]

3.5.2 Been There Done That: Meta-Learning with Episodic Recall [36]

The speaker is Samuel River, joint with Jane Wang.

Consider the Lifelong Learning setting (they call it Meta learning) – interacting with $m \sim D$, an MDP sampled from some distribution.

Goal: Consider the role of reocurrence in meta-learning/lifelong learning.

Keep track of prior tasks:

$$t_n \mid t_1, \ldots, t_{n-1} \sim \Omega(\theta, \mathcal{D}). \tag{38}$$

That is, the next task you sample is conditioned on the prior task you sampled. Framework lets you be precised about task reoccurrence statistics. Additionally get a context, c, when you sample, that tells you whether you’ve seen the task before.

Example: bandits! But actually, contextual bandits, since you also see c.

In general, they make use of an LSTM to solve these kinds of problems.

3.5.3 Transfer in Deep RL using Successor Features in GPI [9]

The speaker is Andre Barreto, joint with Diana Borsa, John Quan, Tom Schaul, David Silver, Matteo Hessel, Daniel Mankowitz, Augustin Zidek, Remi Munos.

Look at a transfer setting: want to transfer knowledge from one task to another.

Their solution:

1. Generalized Policy Improvement (GPI)
2. Successor Features

Generalized Policy Improvement takes as input a bunch of polices, $\pi_1 \dots \pi_n$, and produces them to yield $\tilde{\pi}$ such that:

$$\forall_i : V^{\tilde{\pi}} \geq V^{\pi_i} \tag{39}$$

Successor Features: suppose:

$$R_i = \sum_j w_j R_j, \quad Q_j = \sum_j w_j Q_j. \tag{40}$$

Thus, given a new task, we can apply the successor features and GPIs to quickly yield a good policy for the new task.

....

3.5.4 Continual Reinforcement Learning with Complex Synapses [26]

The speaker is Christos Kaplanis, joint with Murray Shanahan, and Claudia, Clopath.

Goal: Fix catastrophic forgetting in Deep RL.

Synaptic Consolidation Model: Benna Fusi model introduced by Benna and Fusi [10]. Formally:

$$u_1 \leftarrow u1_{+}\eta/C_1\Delta w + g_{1,2}(u_1 - u_2). \tag{41}$$

Summary:

- Consolidation model mitigates catastrophic forgetting in RL agents at multiple timescales.
- Consolidation process is agnostic to timescale of changes in data distribution.

....

4 Thursday July 12th

I made it to the keynote this morning!

4.1 Intelligence by the Kilowatthour

The speaker is Max Welling from Qualcomm.

Alternative talk title: $F = E - H$. That is:

$$\text{Free Energy} = \text{Energy} - \text{Entropy} \tag{42}$$

4.1.1 Free Energy, Energy, and Entropy

In the industrial revolution we changed our ability to perform physical work (energy) – conversely the organizational structures in the world changed, too (entropy).

“It from Bit”:

> It from Bit symbolizes the idea that every item of the physical world has at bottom an immaterial source and explanation... that all things physical are information-theoretic in origin and that this is a participatory universe.
> – John Archibald Wheeler

Erik Velinde: gravity is actually an entropic force.

Free Energy: from physics to informatics:

- Second law of thermodynamics $\Delta H \geq 0$. (Maxwell’s demon!)
- Work $\propto -DeltaF = -(\Delta E - \Delta H)$. Energy is the part of the free energy that cannot be converted into work.

E.T. Jaynes *Information Theory and Statistical Mechanics* [24]:

- The free energy is a subjective quantity.
- Entropy is a degree of ignorance about the microscopic degrees of freedom of a system
- Landauer: computer memory of the microscopic state information needs to be overwritten which increases entropy and costs energy. (Landauer Limit)
- **Takeaway**: modeling is a subjective property.

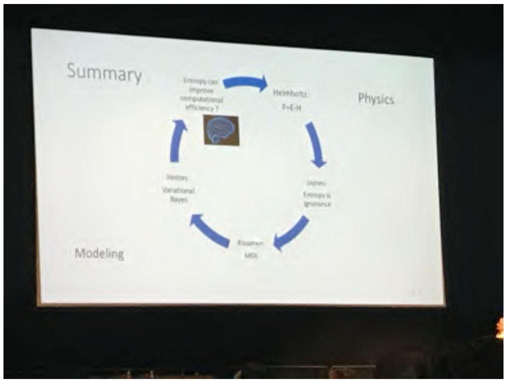
And we get further affirmation of this fact (the above takeaway) from Bayes!

Rissanen: “Modeling by shortest data description.” [35], with $L(\cdot)$ description length:

$$L(\text{Data} \mid H) + L(H) = \underbrace{-\mathbb{E}_{p(\theta|X)}[\log p(X \mid \theta)]}_{\text{bits to encode data}} + \underbrace{KL[p(\theta \mid x) \mid\mid p(\theta)]}_{\text{bits to encode hypothesis}} \tag{43}$$

Then, Hinton drew from these ideas to study simple Neural Networks [21]. Formulation ended up with a similar energy term and entropy term.

To summarize:



Summary

Entropy can improve computational efficiency ?

Helmholtz: F=E-H

Physics

Jaynes: Entropy is ignorance

Rissanen: MDL

Hinton: Variational Bayes

Modeling

Figure 11: Some history of the Free Energy, Energy, and Entropy conversation.

4.1.2 Energy Efficient Computation

A pendulum swing between Vartiational Bayes and MCMC/Sampling based approach. Both have (competing) strengths/weaknesses.

Variational Bayes: Deterministic, Biased, Local Minima, Easy to assess convergence.

Markov Chain Monte Carlo: Stochastic (sample error), Unbiased, Hard to mix between modes, hard to assess convergence.

“The Big Data Dest”: Any reasonable procedure should give you an answer in finite time.

Why think about energy needed for AI now?

1. Value created by AI must exceed the cost to rnu the service
2. AI power and thermal ceiling: as AI moves from cloud to end systems, we need lower energy AI computing.

Main Claim: We should think about the amount of intelligence we get from an AI algorithm per kilowat hour.

One Idea: do model compression via Bayesian deep learning.

Bayesian Compression [32]: we overparameterize neural networks so much—Bayesian compression sparsifies weights really effectively. The sparsfiication is extremely dramatic (some layers went from a few thousand weights to only a handful while retaining the same accuracy). Can offer:

- Compression, quantization
- Regularization, generalization
- Confidence estimation
- Privacy and adversarial robustness

Showed a few other methods of compressing Neural Networks, such as differentiable quantization, spiking neural networks.

End with Steve Jobs:

> This revolution, the information revolution, is a revolution of free energy as well, but of another kind: free intellectual energy.

4.2 Best Paper 2: Delayed Impact of Fair Machine Learning

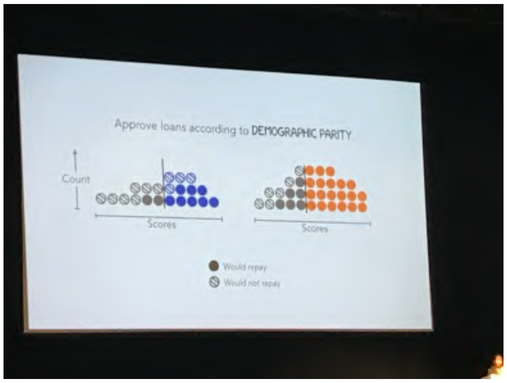
The speaker is Lydia T. Liu, joint with Sarah Dean, Esther Rolg, Max Simchowitz, and Moritz Hardt.

Increasing trend in fairness research: lots of attention, increasing dramatically over time. In all we now have *21 definitions of fairness.*

Usual idea: come up with a definition of fairness such that this definition ensures protected groups are better off. That is, if ML systems are fair, we assume that protected groups are better off.

This paper: is the above assumption correct? How do fair ML systems actually impact protected groups?

Example, loans:



Approve loans according to DEMOGRAPHIC PARITY

Figure 12: Parity in loans.

So: fairness criteria doesn't always help in the relevant sense.

This work:

- Introduce the *outcome curve* a tool for comparing delayed impact of fairness criteria
- Provide a characterization of the delayed impact of 3 criteria.
- Showed that fairness criteria might not always help.

Individuals have scores, $R(X)$, that denote some relevant value for a given domain. If one individual has a score, a group of individuals will have a distribution over scores.

Monotonicity assumption: higher scores imply more likely to repay (in the loan case).

Institution classifies individuals by choosing an acceptance threshold score T to maximize their expected utility.

Main idea behind the failure mode: scores of accepted individuals change depending on their success, sometimes for the worse.

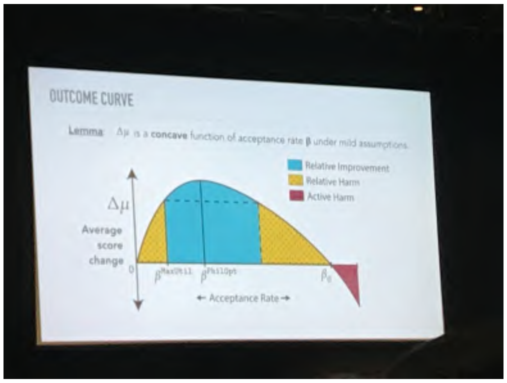
Definition 16 (Delayed Impact): *The delayed impact is:*

$$\Delta\mu - \mathbb{E}[R_{old} - R_{new}]. \tag{44}$$

Lemma 4.1. *$\Delta\mu$ is concave function of acceptance rate β under mild assumptions.*

Theorem 4.2. *All outcomes regimes are possible.*

So: Equal opportunity and demographic parity may cause relative improvement, relative harm, or active harm.



OUTCOME CURVE

Lemma: $\Delta\mu$ is a **concave** function of acceptance rate β under mild assumptions.

Figure 13: Different possible outcomes of fairness measures.

Theorem 4.3. *Demographic Parity may cause active or relative harm by over-acceptance, equal opportunity doesn't.*

Theorem 4.4. *Equal opportunity may cause relative harm by under-acceptance demographic parity never under accepts.*

Run experiments on FICO credit score experiments, the results corroborate their theory, show that the control groups are effected in different ways depending on the metric of fairness.

Dave: I missed a lot of sessions today due to meetings and prepping for my talk. Also I tried entering the Deep Learning Theory session but it was full!

4.3 Reinforcement Learning

Closing with the RL session for the day.

4.3.1 Decoupling Gradient Like Learning Rules from Representations

The speaker is Phil Thomas, joint work Christoph Dann and Emma Brunskill.

This paper: generalized Amari's gradient like-learning rule [2] to naturalized learning rule, including TD-like-algorithms, policy gradient algorithms, and accelerated gradient methods.

4.3.2 PIPPS: Flexible Model-Based Policy Search Robust to the Curse of Chaos [33]

Goal: sample efficient model-based RL based on PILCO.

Three steps:

1. Run control policy $u = \pi(x; \theta)$, gather data D
2. Train dynamics model
3. Optimize policy with model simulations:

Results seem really strong.

5 Friday July 13th

Too much RL to miss!

5.1 Reinforcement Learning

Here we go:

5.1.1 Hierarchical Imitation and Reinforcement Learning [28]

The speaker is Hoang Le, joint with Nan Jiang, Alekh Agarwal, Miroslav Dudk, Yisong Yue, and Hal Daume.

Well known: most RL methods have a hard time learning long horizon and sparse reward tasks (like Montezuma's Revenge).

Definition 17 (Imitation Learning): *A teacher guides a learner by giving demonstrations or feedback indicating how to perform a task (advice usually in the form of near optimal labels/demos).*

Problem: The teacher's feedback can be costly to obtain (lots of good demos are hard to provide). Often depends on the horizon of the problem.

Main Question: How can we most effectively leverage limited teacher feedback?

Alternative types of feedback:

- People like giving high level feedback with hierarchical structure built in.
- People like lazY evaluation

This work:

> Teacher provides high level feedback and zooms in to low level only when needed (savings in teaching effort)Describe a hybrid imitation and RL approach where teacher provides only high-level feedback.

Motivating problem:

- Grid labyrinth with state the image of the grid.
- Primitive actions are up/down/left/right, macro-actions allow the agent to go directly to neighboring rooms.

Key Strategy: Be more selective when choosing to label:

- Don't label if the meta controller chooses incorrect macro-action
- Don't label if subpolicy *accomplished* a correct macro action
- **Only label** if subpolicy fails (that is, the low level execution of the macro-action fails).

Summary: Teacher labels high-level trajectory with correct macro-actions. Key insight is that it's cheaper to verify that low-level trajectory is succssful is chepar than labeling.

Theorem 5.1. *Labeling effort = high level horizon + low level horizon.*

Experimental results on labyrinth tasks show that the labeling approach requires less data to do well than flat imitation learning approaches.

They conclude by extending the algorithm to a hybrid IL/RL case, where they learn the meta-controller by IL and the subpolicies by RL. They test this approach on the first room of Montezuma's, showing that the approach can consistently do well in the first room relative to baselines.

5.1.2 Using Reward Machines for High-Level Task Specification [23]

The speaker is Rodrigo Toro Icarte, joint with Toryn Q. Klassen.

Motivation: Reward functions are extremely difficult to structure in the right way.

Q: How can you exploit the reward function definition, if showed to the agent?

This paper:

1. RMs: a novel language to define reward function
2. QRMs: a new approach for exploiting RMs in RL.

Encode reward functions as a reward machine:

Definition 18 (Reward Machine): *Consists of:*

- *A finite set of states U*
- *An initial state $u_0 \in U$*
- *A set of transitions labelled by:*
  1. *A logical condition on state features.*
  2. *A reward function.*

Then introduce Q-Learning for Reward machines. Idea:

1. Learn one policy per state in the reward machine.
2. Select actions using the policy of the current RM state.
3. Reuse experiences across RMs(?) Dave: I think, I missed it.

Dave: I'm up! [1]

5.1.3 Policy Optimization with Demonstrations [25]

Main focus: Exploration.
Introduce Demonstration-Guided Exploration term:

$$L_M \triangleq D_{JS}(\pi_\theta, \pi_E). \tag{45}$$

5.2 Language to Action

The speaker is Joyce Y. Chai.

Starting with the Jetsons! Comparison to current technology – it seems like we can do most things in the Jetsons, but not Rosei! Why not? Seems like we're still far from Rosie.

Lots of exciting progress: language communication with robots has come extremely far. The next frontier: Interactive Task Learning.

Definition 19 (Interactive Task Learning): *Teach robots new tasks through natural interaction, either through demonstration (visual demo, language or kinsethetic guidance) or through specification (natural language specification, GUI-based specification)*

Demo: person teaching a robot to make a smoothie. The goal is to seamlessly communicate to the robot how to carry out a structured task. The end result is knowledge of a task structure (like a Hierarchical Task Network).

Critical to communication success: common ground (shared context, representation, knowledge, assumptions, abilities, percepts). How can we overcome this problem?

Core questions of the talk:

- Q1: What kind of commonsense knowledge is essential for understanding and modeling action verbs?
- Q2: How do we acquire such knowledge?

5.2.1 Grounding Verbs to Perception

First: how can we understand the semantic role of verbs? Example: Human: [pick up]$_{predicate}$ [a strawberry]$_{ARG}$..

Task: given some commands from a person, we'd like to ground these commands into a semantic representation of some kind (like a grounding graph, grammar – effectively doing semantic parsing from language/sensor input).

For instance, given "She peels the cucumber", we can ask: "what happens to the cucumber?" to determine if the relevant semantics are captured in digesting the initial statement.

Physical causality of action verbs: "Linguistic studies have shown that concrete action verbs often denote some change of state as the result of an action" [22].

Q: Can we explicitly model physical activity?

A: Sure! They collect data in an MTurk like study to annotate verbs with causality knowledge. This enables robotic systems to perceive the environment, observe some knowledge, and apply/extract causal knowledge about the entities of relevance.

The ultimate question we care about, though: “Can the robot perform the action?” → no. Because planning is hard when we have high dimensional inputs as language/images.

5.2.2 Grounding Language to Plans

Q: How do children acquire language?

A: Drawn to the social-pragmatic theory of language acquisition introduced by Tomasello [41]. Main idea: Social interaction is key to facilitating communication, including exercises in intention reading and pattern finding.

Robot learning variant:

1. Learning Phase: language instruction and demo to teach.
2. Execution phase: issue an action command, retrieve best fit representation for action planning/execution, evaluate.
3. (Possibly repeat these two steps).

Use RL to learn an interaction policy – when should the robot ask which questions to maximize long term reward? Implemented this learned interaction policy in a Baxter robot, showed a robot demo where the robot first asks for a demo, clarifies the scene, and then asks to reset the environment and performs the same action itself.

Claim: If robots are to become our collaborators, they must acquire this ability to do action-cause-effect predictions.

Problem: Naive physical Action-Effect Prediction. For example given an action description like “squeeze-bottle”, and a few images showing the consequences of applying that action (and try the reverse)

IMAGE
Their approach: aim to have a small number of annotated examples, then pair these high quality data with simple web search images. Using this approach showed a really nice demo of someone teaching a robot to make a smoothie.

Conclusions:

1. Exciting Journey Ahead!
2. In the pursuit of AGI and Rosie, we still have a long way to go.
3. Upcoming challenges: lots of unknowns, requires a multidsciplinary, joint effort across vision, language, robotics, learning, and more.

4. Things on the wishlist:
   (a) Representations: rich and interpretable.
   (b) Algorithms: interactive and interactive, incorporate prior knowledge, handle uncertainty, and support causal reasoning.
   (c) Commonsense knowledge: cause-effect knowledge including physical, social, and moral.

5.3 Building Machines that Learn and Think Like People

The speaker is Josh Tenenbaum.

Slide one: AI technologies! We have loads of them. But, we don't have any "real AI". We have machines that do things we thought only humans can do, not the kind of flexible general purpose reasoners.

What else do we need? (This talk):

- Intelligence is not just about *pattern recognition* (which has been the focus recently).
- It is about modeling the world:
  1. Explaining and understanding what we see
  2. Imagining things we could see but haven't
  3. Problem solving and planning actions to make these things real.
  4. Building new models as we learn more about the world.

If you want to hear more, check out the work by Lake et al. [27]

Fundamental for MIT's Quest for Intelligence: "imagine if we could build a machine that grows into intelligence the way a person does, that starts like a baby, and learns like a child.

Success means: AI that is truly intelligent, and ML that actually learns.

Early influential/classical papers were published in psych/Cog Sci journals (boltzmann machines paper, finding structure in time, perceptron, etc.).

Now, the science of how children learn, can now offer real engineering guidance to AI. In particular, basic questions:

1. What is the form and content of the starting state (inductive bias)?
2. What are the learning mechanisms?

Turing: "Presumably the child-brain is something like a note-book..."

Cog Sci paper studying how children start to acquire knowledge [39]: in a real sense, they are already born knowing about object permanence and 3d space.

Child as Scientist view: children don't learn by just copying things down. They learn via play (experiments) to test hypotheses actively.

So, fundamental question: how do we grasp onto these ideas in machine learning and AI?

Goal: Reverse-engineering "Core Cognition", intuitive physics, intuitive pscyhology. So, how do we do this?

- Probabilistic programs integrate our best ideas on intelligence: symbolic languages for knowledge representation, composition, and abstraction. Examples: Church, Anglican, WebPPL, Pyro, ProbTorch, etc.
- Probabilistic inference: causal reasoning under uncertainty and flexible inductive bias.
- Neural networks for pattern recognition.

Questions for the rest of the day:

1. Q1: How do these systems work? (above)
2. Q2: How are they learned?

Dave: I had to take off for meetings the rest of the day.

6 Saturday July 14th: Lifelong RL Workshop

In the spirit of "ICRL", I'll be at the lifelong RL workshop (also the topic of both my papers at ICML). I missed the earlier parts of the workshop. First, some orals on multitask RL and meta RL.

6.1 Multitask RL for Zero-shot Generalization with Subtask Dependencies

The speaker is Sungryull Sohn, joint work with Junhyuk Oh and Honglak Lee.

Multitask RL with *flexible* task description: use natural language to provide a seamless way to generalize to unseen complex tasks with compositions. Prior task descriptions focus on single sentence or sequence of instructions.

Motivating Example: Household robot making a meal. One might break it down into subtasks, like: pickup egg, stir egg, scramble egg, pickup bread, and so on.

Instead, one might give high level commands, like "make a meal". But some tasks impose different precondition relations between different subtasks.

This work: decompose subtasks into a graph, then do subtask graph execution problem.

Definition 20 (Multi-task RL): *Let G be a task parameter drawn from a distribution $P(G)$. Here:*

- *G is given to the agent.*
- *G specifies the subtask graph and input observations*
- *The task is defined by G as an MDP tuple: $\langle S, A, R_G, T_G, \gamma_G \rangle$ Dave: and maybe one other component?*

Main idea: construct a differential representation of the subtask graph. Achieved by replacing "AND" and "OR" operations with approximated-and, approximated-or nodes, which are differentiable.

Evaluate in a 2d Minecraft like domain with lots of preconditions (get stone then make stone pickaxe then mine iron etc.).

6.2 Unsupervised Meta-Learning for Reinforcement Learning

The speaker is Abhishek Gupta, joint with Benjamin Eysenbach, Chelsea Finn, and Sergey Levine.

Goal: Fast RL. Current methods take too much time.

Q: What exists in the literature to make RL faster?

11 Model-based RL, 2nd order methods, etc.

22 Using additional supervision, shaping, priors, etc.

33 : Learning from prior experience on related tasks.

Definition 21 (Meta RL): *Learn how to do fast RL from experiment and incorporate prior experience for fast learning. Agent is given prior experience from some set of tasks (with each task an MDP).*

Dave: Is Meta RL a problem or a solution?
Meta-RL requires lots of hand specified task distributions or selection of prior tasks to train on.

This paper: remove this supervision. Yields a general recipe for Unsupervised Meta RL (UMRL). Advantages of UMRL:

- No hand-sepcification needed for meta training
- Fast RL on new tasks

- Less overfitting on task distributions

Q: How can we acquire task distributions without providing supervision?

- One idea: randomly initialize discriminators for reward functions.
- Another idea: use "diversity is all you need idea" to choose tasks that have maximal log likelihood w.r.t. the state. Seems useful as we generate a bunch of new diverse tasks.

Q: How can we learn fast RL algorithms from these tasks?

A: We want: (1) continuous improvement, good extrapolation behavior, (2) reverts to standard RL out of distribution.

MAML: Model Agnostic Meta Learning for RL key idea: learn policy π_θ which can adapt to new tasks with one steps of policy gradient:

$$\max_\theta \sum_{i \in tasks} R_i(\theta_i'). \tag{46}$$

Explore MAML with their unsupervised task generation in Cheetah, Ant, and 2d navigation.

....

7 Sunday July 15th: Workshops

Today I'll be bouncing between the Exploration workshop and the AI for Wildlife Conservation.

7.1 Workshop: Exporation in RL

Three best paper award talks: "Meta-RL of Structured Exploration Strategies"

7.1.1 Meta-RL of Structured Exploration Strategies

Exploration important to: (1) Experience and optimize sparse rewards, and (2) Learn quickly and effectively. Conversely, humans perform highly directed exploration.

Main Q: Can we use our prior experience to learn better exploration strategies?

Problem: Given some prior experience on tasks $(T_0, \ldots, T_n) \sim p(T)$, with each task an MDP. Then, on some new test task T_{test}, we'd like the agent to learn/make good decisions as quickly as possible.

Two key insights:

1. Explore in the space of random but structured behaviors.

2. Quickly adapt behavior to new tasks once rewards are experienced.

Q: How do we generate coherent exploration behavior?

Idea: Use structured stochasticity. In particular: noise in latent space generates directed temporally coherent behaviors. Exploring with noise in latent space.

Then, do Meta-Training with "MAESN". That is: train a latent policy π_θ across multiple tasks, each with some parameter. Then optimize the meta-objective while constraining pre-update latent parameters against a prior.

At test time: do RL where you initialize the latent distribution based on the prior.

7.1.2 Counter-Based Exploration with the Successor Representation

The speaker is Marlos Machado, joint with Marc Bellemare and Michael Bowling.

The Successor Representation [14].

Formally speaking:

$$\psi_\pi(s, s') = \mathbb{E}_{\pi,p}\left[\sum_{t=0}^{\infty} \gamma^t \nVdash \{s_t = s' \mid s_0 = s\}\right] \tag{47}$$

This work: stochastic successor representation. Compute empirical model of the stochastic SR:

$$\tilde{P}_\pi(s, s') = \frac{n(s, s')}{n(s) + 1}. \tag{48}$$

Idea: let's us count state visitation, which we can use for exploration.

Similar to return:

$$\tilde{\psi}_\pi(s_1) = \frac{1}{n(s) + 1} + \dots \frac{1}{n(s_k) + 1)}. \tag{49}$$

Let's them introduce an exploration bonus using these state-visitations.

Run some experiments in River Swim and compare to usual PAC-MDP algorithms, performs competitively to R-Max, E^3, and so on.

Main reason to do this, though, is that the successor representation can easily generalize to function approximators.

7.1.3 Is Q Learning Provably Efficient

The speaker is Chi Jin, joint with Zeyuan Allen-Zhu, Sebastien Bubeck, and Michael I. Jordan.

Main Q: Can model-free algorithms be made sample efficient? In particular, is Q-Learning provably efficient?

First, what *do* we know about learning in tabular MDPs?

Main result shows that Q-Learning with UCB like exploration strategy has bounded regret:

$$\tilde{O}\left(\sqrt{H^4 SAT}\right). \tag{50}$$

Which is competitive with model-based regret bounds (like UCRL).

Some other insights:

- Set learning rate to H/t, with H the horizon and t the state visitation for that state.
- If you vary the learning rate, you can prioritize earlier vs. later updates, which also unturns a bias variance trade-off.

7.1.4 Upper Confidence Bounds Action-Values

The speaker is Martha White, joint with her students (whose names I missed – sorry!).

Goal: Discuss on direction for UCB on action-values in RL, highlight some open questions and issues.

Problem setting:

- General state/action space.
- Agent estimates action-values from stream of interaction.
- How can the agent be confident in its estimates of $Q^*(s, a)$.
- Our goal: directed exploration to efficiently estimate $Q^*(s, a)$.

Many model-free methods use uncertainty estimates: (1) Estimate uncertainty in $Q(s, a)$, and (2) Reward bonuses or pseudo-counts. Let's talk about (1)

In stochastic bandits, a bit more clear how to compute our UCB. Same story, roughly, in contextual bandits – we can still compute UCB like estimates in this setting.

Q: Why is RL from the contextual bandit setting?

A1: Temporal connections. A2: Bootstrapping – do not get a sample of the target, especially since the policy is changing.

Idea for UCB in RL: UCB for a fixed policy. Apply our usual concentration inequalities to obtain the relevant upper bound w.r.t. the chosen fixed policy.

To extend this to an unfixed policy – using some ideas from Ian Osband and Ben Van Roy's work on stochastic optimism can enable the right sort of guarantees.

Empirically, algorithms that use this kind of algorithm seem to work quite well: (1) Bootstrap DQN, (2) Bayesian DQN, (3) Double Uncertain Value Networks, (4) UCLS (new algo in this work).

Conduct experiments in a continuous variant of the River Swim domain.

UCLS and Bayesian DQNs can both incorporate some of these ideas seamlessly in the neural case by modifying the last layer of an NN.

Open Questions:

- Do we have to estimate UCB directly on Q^*?
- Can we estimate UCB on Q^* and iterate?
- Is it useful to use UCB derived for fixed policies, but with inflated estimates of variance to get stochastic optimism?

7.2 Workshop: AI for Wildlife Conservation

Focus: Unprecedented change in biodiversity and species loss:

1. Humans have increased the species extinction rate by as much as 1000 times over background rates.
2. 10-30% of mammals and birds are facing extinction.

AI can help! Examples: predicting species ranges, migrations, poaching activity, planning ranger patrolling and conservation investments, detecting species, and so on.

7.2.1 Data Innovation in Wildlife Conservation

The speaker is Jennifer Marsman, from MSR's AI for Earth group.

Focus: Conservation groups have a scale problem.

Microsoft's AI for Earth initiative[3]:

- *Agriculture:* to feed the world's growing population, farmers must produce more food on less arable land with less environmental impact.
- *Water:* In less than two decades demand for fresh water is projected to outpace supply.
- *Biodiversity:* Species going extinct beyond the natural rate by orders of magnitude.
- *Climate Change.*

Main focus for today: innovations in data at scale.

Some examples:

[3] `www.microsoft.com/AIforEarth`

1. *Tagging:* Orca killed by satellite tag leads to criticism of science practices.

   → In trying to measure we harm rather than help.

2. *Conservation:* poachers are using data from wildlife scientists to target and kill rare species.

   → Data can be used in malicious and unexpected ways.

3. (and more)

The above summarize a few concerns around data. So, in this talk: how can we innovate our attitude toward data? Five suggestions: (1) UAV/Drone imagery, (2) Camera trap, (3) Simulation, (4) Crowd Sourcing, (5) Social Media.

UAV/Drone Imagery

Consider the FarmBeats challenge, issued by Microsoft. Goal is to provide farmers with access to Microsoft Cloud and AI technologies, enabling data-driven decisions to help farmers improve agricultural yield, lower costs, and reduce the environmental impact.

→ The challenge: by 2050, the demand for food is expected to outpace production by over 70%.

Solution: FarmBeats uses ML to integrate sensor data with aerial imagery to deliver actionable insights to farmers, all at a fraction of the cost of existing solutions. Developed an app to help farmers automate the tasks of drones to bolster the effectiveness of their farm.

Q: How can we offer connection to remote areas?

Idea: TV White spaces– use *unoccupied* TV channels to send wireless data. TV uses lower frequencies so they reach quite far (also part of the FarmBeats project).

Camera Trap Data

Challenges:

- Lighting, angles, occlusion, etc.
- Fairly consistent background may make it easier to learn noise.
- Unbalanced dataset (animals don't line up for photos).
- Video vs. still images
- Motion-triggered, heat triggered, or time-based?

Simulation

Example: used a simulation as part of a challenge problem in the workshop focused on flying UAVs and drones. Lots of opportunities to use simulations!

Crowdsourcing

AI for Earth has been working with iNaturalist to crowdsource data on biodiversity.

Idea: when you're out taking a nature walk, you can contribute data to a large publiclly available dataset for folks working on biodiversity.

AI for Earth offers some pre-trained methods to help out with specifies identification, uses an existing dataset of animals and also geographic information to help gather good data. Avoids the bottleneck of novice users needing to know different species.

Social Media for Biodiversity Data

Wildbook system uses ML to find exact animals (not just species, but literally the same exact animal). Wilfbook has an agent that scans social media for images of different animals to track the location of individual animals. Thus: they can track the migration of a specific whale. (Main ML idea is to use distinct SIFT-like-markers that pick out specific animals). Available here: `https://www.whaleshark.org/`
Next up are spotlight talks from papers.

7.2.2 Computing Robust Strategies for Managing Invasive Paths

The speaker is Marek Petrik, joint with Andreas Luydakis, Jenica Allen, and Tim Szewczyk.

Focus: Invasive species, in particular, the glossy buckthorn.

Problem: Glossy buckthorn is responsible for both ecological and financial damage all over.

Solution: Optimize where, when, and how to target glossy buckthorn (burn? cut? etc.).

Important that recommendations are high confidence/correct. These actions are expensive and have long term consequences. Previous approaches are heuristic and just make best guess.

Even further difficulty: Ecological data is often extremely sparse and biased. For instance: most reports of glossy buckthorn appear next to roads—obviously a result of sampling bias!

Goal: reliable data-driven methods for this problem.

Their approach: robust optimization. A method that can be used to incorporate confidence into predictions. IDea: take a point estimate and replace it with a set of plausible realizations, which yields a mini-max problem:

$$\max_{\text{allocation}} \min_{\text{presence}} \text{benefit}(\text{allocation}, \text{presence}). \tag{51}$$

Run simulations based on real data from EDDMaps and WorldClim.

Summary:

1. Robustness matters when making decisions in critical domains.
2. Reeal-world data is limited, biased, sparse.
3. Robust optimization is a tractable approach that can account for uncertainty in predictions.

7.2.3 Detecting and Tracking Communal Bird Roosts in Weather Data

The speaker is Daniel Sheldon.

Focus: Bird Migration data. Huge dataset! In particular targeting bird roosts where birds commune and fly in a specific location for long periods of time.

Data set: detailed biological phenomenon, 143 radar stations, more than 200 million entries.

Goal: Develop an automated system to detect and track bird information in this data set.

Tracking system provides:

- Quantitative info about distributions, movements.
- Basic knowledge about the birds and their communities.

In the end, created an annotated dataset of tree swallow roosts and their movements in the US.

7.2.4 Recognition for Camera Traps

The speaker is Sara Beery, joint with Gran Van Horn and Pietro Perona.

Camera Traps: motion or heat sensor cameras that take photos of wildlife, yielding both presence and absence of animals (which can give better population estimates). Can even get a short sequence of images that shed light on movement, depth.

Problems: (1) flash can effect animals, (2) cameras often triggered by nothing (wind, people), (3) Data is hand sorted by experts (costly!), so even if cameras get cheaper, we can't scale due to the labeling.

Huge challenges with the data:

1. Illumination
2. Blur
3. ROI Size
4. Occlusion

5. Camouflage
6. Perspective

This work: organize the data set by location[4], animal type, bounding boxes, and so on.

What else can we do with camera traps?

- Novelty detection
- Direction of travel
- (and more!)

7.2.5 Crowdsourcing Mountain Images for Water Conservation

The speaker is Darian Frajberg, joint with Piero Fraternali and Rocio Nahime Torres.

Environmental monitoring is receiving a huge boost from mobile crowdsourcing (per the methods outline in the previous few talks and the keynote).

This work: "SnowWatch". Create novel and lost cost tools to monitor and predict water availability in dry season in mountainous regions.

Mobile applications for crowd-sourcing already exist, but not for mountains.

Given the success of Pokemon Go, they use an Augmented Reality method.

Build an application called PeakLens for Android: an outdoor Augmented Reality app that identifies mountain peaks and overlays them in real-time view.

Main technical challenge is to identify the mountain range from the skyline – difficult due to occlusions, compass/GPS error, and low resolution images. Achieve extremely high accuracy (90%+).

7.2.6 Detecting Wildlife in Drone Imagery

The speaker is Benjamin Kellenberger, joint with Diego Marcos and Devis Tuia.

Dataset: Kuzikus dataset, contains drone imagery of large animal species like Rhino, Ostrich, Kudu, Oryz, and so on. Around 1200 animals in 650 images.

Problem: the animals take up a very small percentage of pixels. Dataset is extremely heterogeneous, finding animals poses a needle-in-the-haystack problem.

Core contribution of the work: new insights to encourage a CNN to train effectively on a dataset presenting the above challenges (animals are very small, similar background across images, and so

[4] `beerys.github.io`

on).

Leverage: (1) Curriculum learning, (2) Border classes (can help identify rare animals using things like animal shadows), and a few other techniques. Together, they complement one another to yield a practically useful detection algorithm for this animal image dataset.

Dave: And that's a wrap!

....

References

[1] David Abel, Dilip Arumugam, Lucas Lehnert, and Michael L. Littman. State abstractions for lifelong reinforcement learning. In *Proceedings of the International Conference on Machine Learning*, 2018.

[2] Shun-Ichi Amari. Natural gradient works efficiently in learning. *Neural computation*, 10(2): 251–276, 1998.

[3] Sanjeev Arora, Rong Ge, Yingyu Liang, Tengyu Ma, and Yi Zhang. Generalization and equilibrium in generative adversarial nets (gans). *arXiv preprint arXiv:1703.00573*, 2017.

[4] Sanjeev Arora, Nadav Cohen, and Elad Hazan. On the optimization of deep networks: Implicit acceleration by overparameterization. *arXiv preprint arXiv:1802.06509*, 2018.

[5] Sanjeev Arora, Rong Ge, Behnam Neyshabur, and Yi Zhang. Stronger generalization bounds for deep nets via a compression approach. *arXiv preprint arXiv:1802.05296*, 2018.

[6] Kavosh Asadi, Dipendra Misra, and Michael L Littman. Lipschitz continuity in model-based reinforcement learning. *arXiv preprint arXiv:1804.07193*, 2018.

[7] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. *arXiv preprint arXiv:1802.00420*, 2018.

[8] Simon Baron-Cohen, Alan M Leslie, and Uta Frith. Does the autistic child have a theory of mind? *Cognition*, 21(1):37–46, 1985.

[9] Andre Barreto, Diana Borsa, John Quan, Tom Schaul, David Silver, Matteo Hessel, Daniel Mankowitz, Augustin Zidek, and Remi Munos. Transfer in deep reinforcement learning using successor features and generalised policy improvement. In *International Conference on Machine Learning*, pages 510–519, 2018.

[10] Marcus K Benna and Stefano Fusi. Computational principles of synaptic memory consolidation. *Nature neuroscience*, 19(12):1697, 2016.

[11] Ronen I Brafman and Moshe Tennenholtz. R-max-a general polynomial time algorithm for near-optimal reinforcement learning. *Journal of Machine Learning Research*, 3(Oct):213–231, 2002.

[12] Robert Calderbank, Sina Jafarpour, and Robert Schapire. Compressed learning: Universal sparse dimensionality reduction and learning in the measurement domain. *preprint*, 2009.

[13] Will Dabney, Georg Ostrovski, David Silver, and Rémi Munos. Implicit quantile networks for distributional reinforcement learning. *arXiv preprint arXiv:1806.06923*, 2018.

[14] Peter Dayan. Improving generalization for temporal difference learning: The successor representation. *Neural Computation*, 5(4):613–624, 1993.

[15] Maria Dimakopoulou and Benjamin Van Roy. Coordinated exploration in concurrent reinforcement learning. *arXiv preprint arXiv:1802.01282*, 2018.

[16] Ronen Eldan and Ohad Shamir. The power of depth for feedforward neural networks. In *Conference on Learning Theory*, pages 907–940, 2016.

[17] Mehrdad Farajtabar, Yinlam Chow, and Mohammad Ghavamzadeh. More robust doubly robust off-policy evaluation. *arXiv preprint arXiv:1802.03493*, 2018.

[18] Rong Ge, Furong Huang, Chi Jin, and Yang Yuan. Escaping from saddle pointsonline stochastic gradient for tensor decomposition. In *Conference on Learning Theory*, pages 797–842, 2015.

[19] Rong Ge, Jason D Lee, and Tengyu Ma. Matrix completion has no spurious local minimum. In *Advances in Neural Information Processing Systems*, pages 2973–2981, 2016.

[20] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[21] Geoffrey E Hinton and Drew Van Camp. Keeping the neural networks simple by minimizing the description length of the weights. In *Proceedings of the sixth annual conference on Computational learning theory*, pages 5–13. ACM, 1993.

[22] Malka Rappaport Hovav and Beth Levin. Reflections on manner/result complementarity. *Syntax, lexical semantics, and event structure*, pages 21–38, 2010.

[23] Rodrigo Toro Icarte, Toryn Klassen, Richard Valenzano, and Sheila McIlraith. Using reward machines for high-level task specification and decomposition in reinforcement learning. In *International Conference on Machine Learning*, pages 2112–2121, 2018.

[24] Edwin T Jaynes. Information theory and statistical mechanics. *Physical review*, 106(4):620, 1957.

[25] Bingyi Kang, Zequn Jie, and Jiashi Feng. Policy optimization with demonstrations. In *International Conference on Machine Learning*, pages 2474–2483, 2018.

[26] Christos Kaplanis, Murray Shanahan, and Claudia Clopath. Continual reinforcement learning with complex synapses. *arXiv preprint arXiv:1802.07239*, 2018.

[27] Brenden M Lake, Tomer D Ullman, Joshua B Tenenbaum, and Samuel J Gershman. Building machines that learn and think like people. *Behavioral and Brain Sciences*, 40, 2017.

[28] Hoang M Le, Nan Jiang, Alekh Agarwal, Miroslav Dudík, Yisong Yue, and Hal Daumé III. Hierarchical imitation and reinforcement learning. *arXiv preprint arXiv:1803.00590*, 2018.

[29] Lisa Lee, Emilio Parisotto, Devendra Singh Chaplot, Eric Xing, and Ruslan Salakhutdinov. Gated path planning networks. *arXiv preprint arXiv:1806.06408*, 2018.

[30] Jan Leike. Nonparametric general reinforcement learning. *arXiv preprint arXiv:1611.08944*, 2016.

[31] Roi Livni, Shai Shalev-Shwartz, and Ohad Shamir. On the computational efficiency of training neural networks. In *Advances in Neural Information Processing Systems*, pages 855–863, 2014.

[32] Christos Louizos, Karen Ullrich, and Max Welling. Bayesian compression for deep learning. In *Advances in Neural Information Processing Systems*, pages 3288–3298, 2017.

[33] Paavo Parmas, Carl Edward Rasmussen, Jan Peters, and Kenji Doya. Pipps: Flexible model-based policy search robust to the curse of chaos. In *International Conference on Machine Learning*, pages 4062–4071, 2018.

[34] Neil C Rabinowitz, Frank Perbet, H Francis Song, Chiyuan Zhang, SM Eslami, and Matthew Botvinick. Machine theory of mind. *arXiv preprint arXiv:1802.07740*, 2018.

[35] Jorma Rissanen. Modeling by shortest data description. *Automatica*, 14(5):465–471, 1978.

[36] Samuel Ritter, Jane X Wang, Zeb Kurth-Nelson, Siddhant M Jayakumar, Charles Blundell, Razvan Pascanu, and Matthew Botvinick. Been there, done that: Meta-learning with episodic recall. *arXiv preprint arXiv:1805.09692*, 2018.

[37] Pouya Samangouei, Maya Kabkab, and Rama Chellappa. Defense-gan: Protecting classifiers against adversarial attacks using generative models. *arXiv preprint arXiv:1805.06605*, 2018.

[38] Sven Schmit and Ramesh Johari. Learning with abandonment. In *International Conference on Machine Learning*, pages 4516–4524, 2018.

[39] Elizabeth S Spelke, Karen Breinlinger, Janet Macomber, and Kristen Jacobson. Origins of knowledge. *Psychological review*, 99(4):605, 1992.

[40] Masanori Suganuma, Mete Ozay, and Takayuki Okatani. Exploiting the potential of standard convolutional autoencoders for image restoration by evolutionary search. *arXiv preprint arXiv:1803.00370*, 2018.

[41] Michael Tomasello. Beyond formalities: The case of language acquisition. *The Linguistic Review*, 22(2-4):183–197, 2005.

[42] Yunbo Wang, Zhifeng Gao, Mingsheng Long, Jianmin Wang, and Philip S Yu. Predrnn++: Towards a resolution of the deep-in-time dilemma in spatiotemporal predictive learning. *arXiv preprint arXiv:1804.06300*, 2018.

[43] Nevan Wichers, Ruben Villegas, Dumitru Erhan, and Honglak Lee. Hierarchical long-term video prediction without supervision. *arXiv preprint arXiv:1806.04768*, 2018.

[44] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

[45] Yingce Xia, Xu Tan, Fei Tian, Tao Qin, Nenghai Yu, and Tie-Yan Liu. Model-level dual learning. In *International Conference on Machine Learning*, pages 5379–5388, 2018.

[46] Andrea Zanette and Emma Brunskill. Problem dependent reinforcement learning bounds which can identify bandit structure in mdps. In *International Conference on Machine Learning*, pages 5732–5740, 2018.

[47] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530*, 2016.