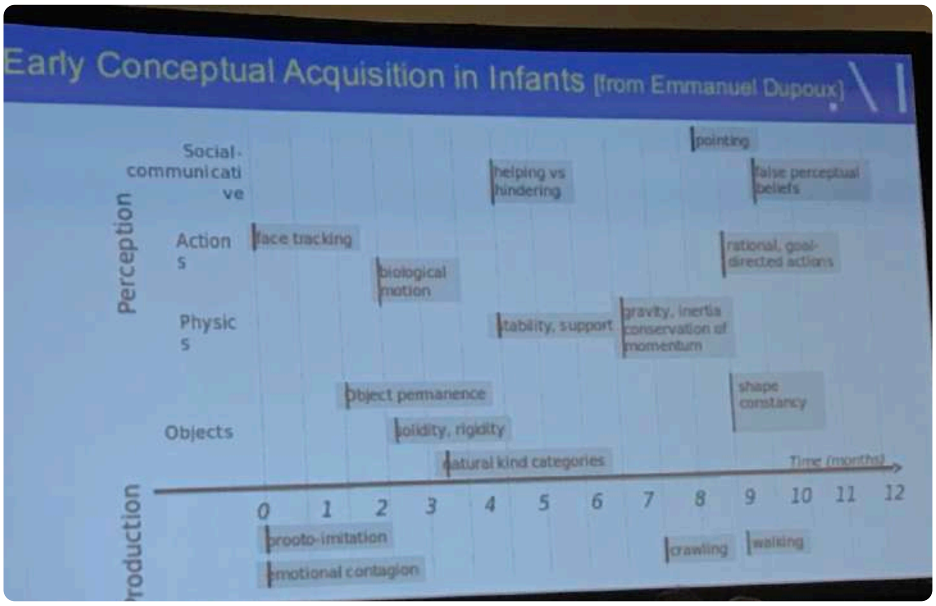
NIPS – DAY 5 Workshops I – Deep Reinforcement Learning

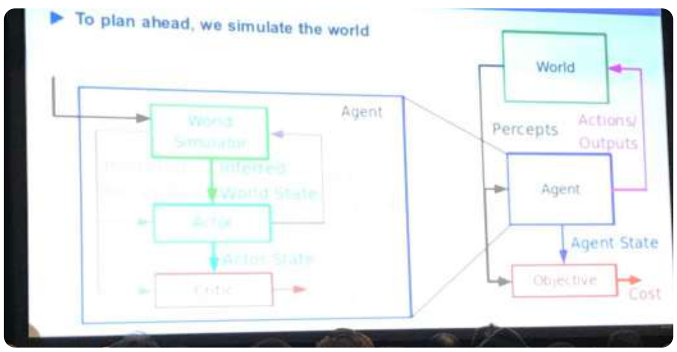
① Y. LeCun – Learning Predictive Models

- Sample complexity problem of model-free RL ⇒ only viable in fast sim. envs.
- Human learning → internal model of env dynamics ⇒ E. Dupoux: Babies learn mainly from observation, little interaction! → reaction time experiments



- Possible solution: Self-Supervised Learning
  - → predict any part of input from any other part
    - ↳ occluded from visible
    - ↳ accumulation of background knowledge of the world!
  - → requires a lot of feedback / trials!

- 'Filling the blanks' ⇒ what the brain does all the time
  - → prediction as essence of intelligence



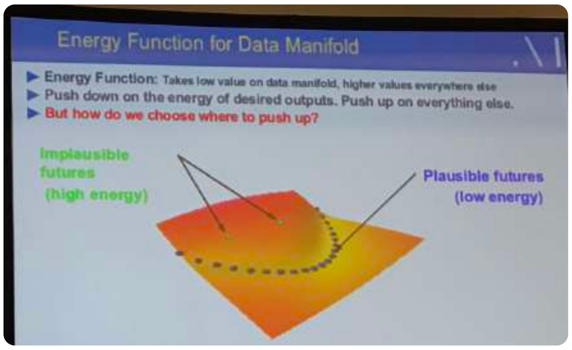
→ adjoint state method = backprop through time (control theory)

→ Forward model training
- ↳ PhysNet ICML 2016
- ↳ blurry prediction when uncertainty
  - ⇒ need to uncertainty-weight average!

→ Solution: Latent Var. Energy-based Models!
- ↳ GANs ⇒ need distance fct. to deter. manifold
- ↳ Regularized Latent Var EBM ⇒ optimize latent using Euclidean dist.
  - ↳ but need to restrict latent var.!

high energy ≈ high uncertainty

- Video prediction → blurriness ⇔ uncertainty
  - ↳ combine with semantic segmentation
  - ↳ Mask-RCNN ⇒ moving masks



- Stochastic latent forward models
  - → don't optimize latent, but predict
    - ↳ latent var corrects residual error
  - → train policy network by prop. through cost function ⇒ added uncertainty cost in objective function (MPUR)
    - ↳ fail ⇒ divergent idea!

→ no interaction with real world needed ⇒ only observations

2a One-shot high fidelity imitation (DeepMind)

- Over-imitation ⇒ behaviour overfitting → kids but not chimps
  - → psychologist belief that human behavior is used to learn new tasks quickly ⇒ seems different but is not!
- Def. of imitation reversed!
  - → use larger networks; before smaller ones because belief that variance in gradients too high!

2b Robotic Manipulation through Visual Planning and Acting (Berkeley)

- Rope tying ⇒ visualization of action sequences
- Info GAN ⇒ noise sampled from meaningful latent representation
  - ↳ causal version ⇒ causal latent representation → second latent vector
- Visual plan → inverse model for action finding
- Added moving obstacles → context-conditional CIGAN

~~AMAZING~~ WORK!

2c Contingency-Aware Exploration in RL (Michigan)

- State representation learning for exploration ⇒ not only visual similarity
  - → Contingency awareness: recognition that some aspect of env dynamics are under agent's control! ⇒ self-location in latent space + awareness
- Attentive dynamics model with spatial attention → contingent region discovery
- Exploration-count based with embedding learning

③ 5. Andreas – Linguistic scaffolds for policy learning (Berkeley)

- Text instruction of tasks in interactive environments
  - → rational speech acts model ⇒ guesses / hints → Linguistic Pragmatics
    to quantification with rewards
    ⇒ Language use ± Gameplay → get away from supervised NLP paradigm
- NLP implications:
  ① Train base listener model ⇒ discr.
  ② Train speaker model that acts when interacting with base → generator
  ↳ captions / segdseg / instruction following / generation
- Problem of overfitting game → does not generalize to real language
  → self-play / tree search ⇒ within rules of language

↑ RL for language ↑

↓ Language for RL ↓

- learning with policy sketches ⇒ Verbalizability of task as skill / context!
- Options → Andreas, Klein & Levine (2016) ⇒ only top-level supervision via sketches → modular composition of sketch policies!
- Importance of generalization! → structured language gives a lot of help
  - Learning language and acting simultaneously
    ↳ form of multi-task learning!
    ↳ interpretability!

Connects to Herzi!

④ S. Kakade – Curiosity and Maximum Entropy Exploration

- RL: algorithms > architectures ⇒ importance of saddle points
  → exploration via intrinsic motivation → Singh et al (2005)
- Policy implies induced distribution over states → capability of agent's control
  → Might want high entropy/ distribution
  uniform
  → don't optimize reward but entropic measures

$$d_\pi(s) = (1-\gamma) \sum_{t=1}^{\infty} \gamma^t P(s_t = s \mid \pi)$$

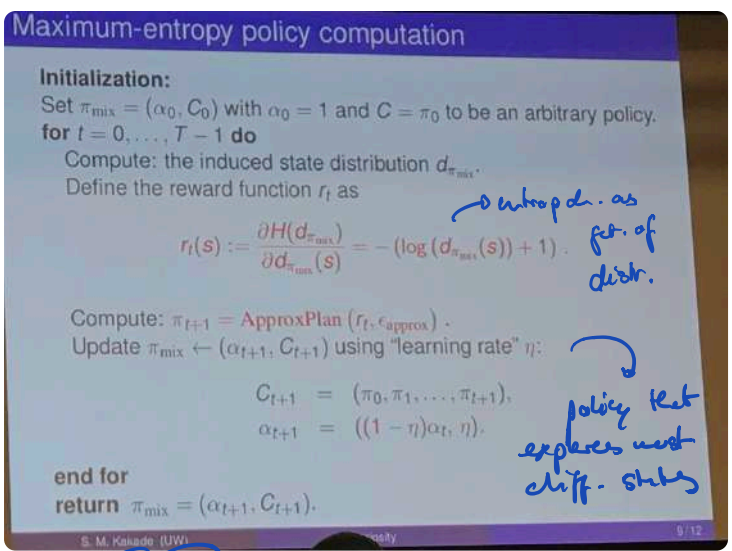
→ converge of state space!

$$H(d_\pi) = -\mathbb{E}_{s \sim d_\pi} \log d_\pi(s) \Rightarrow \pi^* = \arg\max_\pi H(d_\pi)$$

↳ not concave in π → How to optimize?

- Mixture over distr.: induces stochasticity for optimization

$$d_{\pi_{mix}}(s) = \sum_i \alpha_i d_{\pi_i}(s)$$



Known MDP-Case

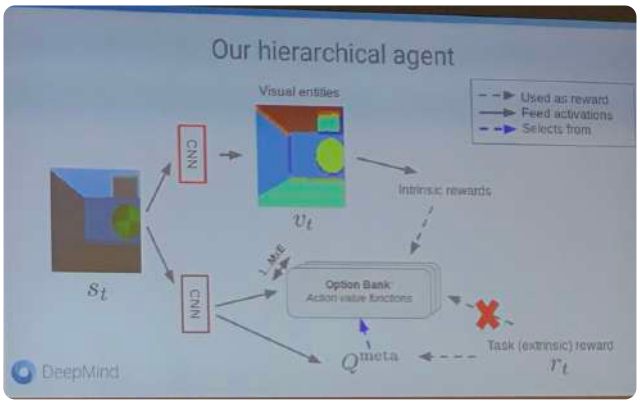
→ Provide proofs!

→ the known MDP ⇒ KL-div.

↳ use other rewards than simply reward!

→ Perspective: Exploration from an optimization pov

(5a) Learning to Control Visual Abstractions for Structured Exploration (DeepMind)



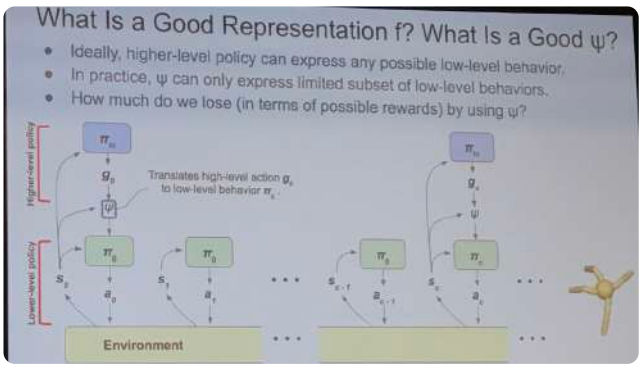
- Visual entities via masks extracted by CNN
  - ↳ segmentation ⇒ contrastive loss
- Extrinsic rewards ⇒ geometric
  - ↳ object controls ? ⇒ unsupervised
  - ↳ not only discovering but different domains identification

(5b) Folding Zero : Protein Folding from Scratch (Alphafold)

- Thermodyn. Hypothesis ⇒ Hydrophobic-Polar Model

(5c) Near-optimal Representation Learning for HRL (Google)

- Goal-conditioned HRL ⇒ high-level actions = goals ⇒ have to be achieved by lower level
- Problem of defining state representation function $f(\cdot)$
  - ↳ prove near-optimal of construction



$$\text{SubOpt}(\psi) = \sup_{s \in \mathcal{S}} V^{\pi^*}(s) - V^{\pi^*_{hier}}(s)$$

- ↳ objective fct. with annealing fct. approx.
- ↳ empirical comparison to oracle

⑥ Deep Recap – Knowledge Representation for RL

- Want agent that continually grows its knowledge → no catastrophic forgetting!

→ Which representation?

→ building blocks for knowledge

$\mathbb{E}\left[\sum_{t=0}^{\infty} R_{t+1} \prod_{i=t+1}^{k} \gamma \mid S_0 = s, t_{\text{term}}\right]$ ← Value fct.

Generalized Value Function!

restrictive: task, discount, reward!

relation to UVFA!

relation to successor feature ↔ Barreto et al '17, '18

Knowledge representation: Generalized Value Functions (GVFs)

- Given a cumulant function c, state-dependent continuation function γ and policy π, the Generalized Value Function $v_{\pi,\gamma,c}$ is defined as:

$$v_{\pi,\gamma,c}(s) = \mathbf{E}\left[\sum_{k=t}^{\infty} C_{k+1} \prod_{i=t+1}^{k} \gamma(S_i) | S_t = s, A_{t:\infty} \sim \pi\right]$$

- *Cumulant* c is a function that can depend on the S_t, or (S_t, A_t) or (S_t, A_t, S_{t+1}) and can output a vector (even a matrix)
  Eg $C_{k+1} = c(S_k, A_k, S_{k+1})$
- *Continuation function* γ maps states to [0,1]
- Cf. Horde architecture (Sutton et al, 2011); Adam White's thesis; inspiration from Pandemonium architecture

DeepRL workshop 2018

parallel prediction computation

} more flexible, can easily get classic value fct.

→ linearity allows us to combine different VFs!

Recently Cumulant + Continuation function

Successor states and successor features are GVFs

- *Successor features* (Barreto et al, 2017, 2018) are a natural extension of successor states (Dayan, 1992)
- If states are defined by a feature vector $\phi(s)$, successor features give the discounted sum of future feature vectors from a state.
- In GVF terms, the cumulant is $c = \phi$, and there is a fixed policy and discount
- Interesting property highlighted in Barreto et al:

$$v_{\mathbf{w}^T c,\pi,\gamma}(s) = \mathbf{w}^T v_{c,\pi,\gamma}(s)$$

which leads to very efficient use of existing GVFs to do one-shot computation of new GVFs

→ think not only from execution point of view!
↳ model ⇒ predictions about future!

Option models

- *Option model* has two parts:
  1. *Expected reward* $r_\omega(s)$: the expected return during the execution of option ω from s
     - Needed because it is used to update the agent's internal representations
  2. *Transition model* $P_\omega(s'|s)$: a sub-probability distribution over next states (reflecting the discount factor γ and the option duration) given that ω executes from s
     - P specifies *where* the agent will end up after the option/program execution and *when* termination will happen
- Models are *predictions* about the future, conditioned on the option being executed

The anatomy of the reward option model

- Option reward model is defined as:

$$r_\omega(s) = \mathbf{E}[R_{t+1} + \gamma R_{t+2} + \ldots | S_t = s, \omega_t = \omega]$$

We can write Bellman equations for the model:

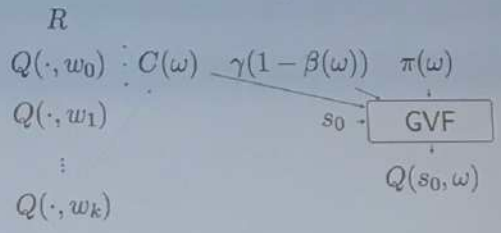
$$r_\omega(s) = \sum_a \pi_\omega(s,a)[r(s,a) + \sum_{s'} P_\omega(s'|s)\gamma(1-\beta_\omega(s'))r_\omega(s')]$$

- This means the option reward model is a GVF, $v_{\omega,r}$: cumulant is the environment reward r, policy is π_ω, continuation function is $\gamma(1-\beta_\omega)$
- Note that *we could use* $\gamma = 1$ *for the overall task* and these GVFs would still be well defined
- Option transition model can similarly be shown to be a GVF

Option formulation "drops" things into pieces

Option as generalized value fct.

GVF for option-critic



- Option-value functions try to do a lot! Which explains some of the pathological behaviro observed in option-critic
- Models of options insulate experience within the option
- Intuitively, models should be easier to both learn and use than option-value functions

→ Can also rewrite policy gradient fct.!
↳ mix & match ⇒ true policy

⇒ GVF allows for easy combination of algorithms → LEGO-ISE

↳ Alternative: Recognizers
⇒ construct target policy that aligns with behavior!

↳ longer / combined learning effect

↳ make off-policy more efficient ⇒ variance reduction effect!
↳ don't update at every timestep!

To do list

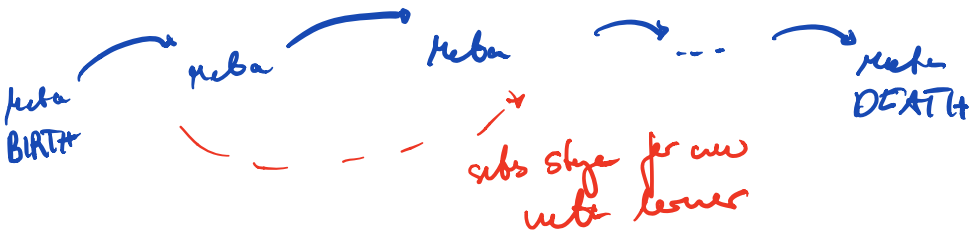
- How do we generate behavior from GVFs ?
  - Option call-and-return execution gives a hint, but is too limiting
  - Ideally, we will use GVFs to plan
  - Exploration: how to generate data for many GVFs?
- Discovery: how do we get initial conditions, policies and continuation functions?
  - Akin to the option discovery problem, but with an emphasis on building models
  - How do we get cumulants?

- HRL environments lacks
  → structure in env
  → cont. state-action spaces where good solutions are available in discretized version

- Can value fct. capture everything relevant to build RL solutions?
  → lower expectation operator ⇒ Hypothesis tests can be tested!

J. Schmidhuber — Curiosity, Lifelong Meta-Learning & Artificial Curiosity

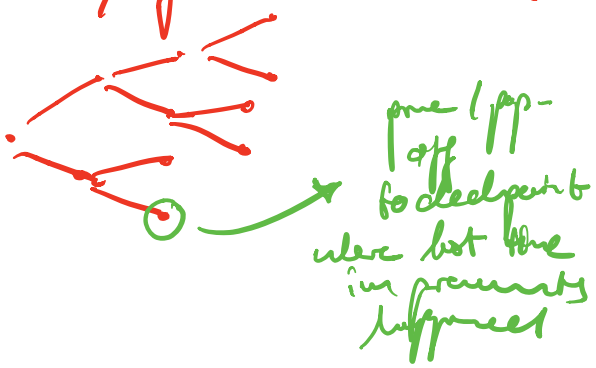
- Meta-learning + Transfer learning → not adjustments of few parameters
  ↳ self-referential, self-modifying ⇒ recursive self-improvement
  ↳ limits of computability and physics
- critic of Mechanicalism ⇒ Old 1994 work → probabilistic policy that changes itself!



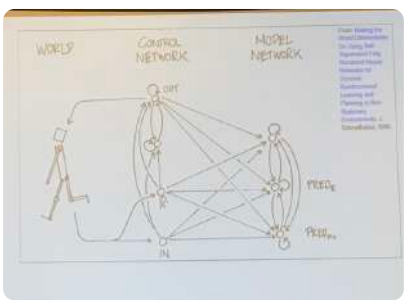
continual ≈ single life!

SUCCESS STORY ALGO → set check points of self-modification ⇒ history of reward accelerations

↳ 'Is life better before or after checkpoint?'



- Gödel Machine (2003) ⇐ self-proving
- RNN Controller & World Model → controller learns ability to learn the model → how to motivate



- Artificial curiosity → adversarial minimax approach
  ↳ improvements from "play-ful" experiments
  ↳ enhanced GANs reference

Hyperparameter

↓

Prior

↓

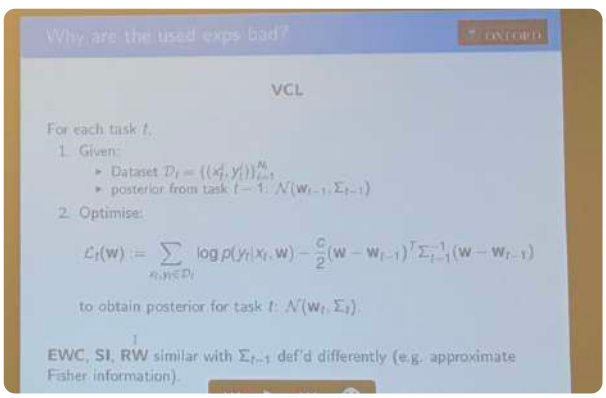
Posterior

(8) Y. Gal — Towards Robust Evaluations of Continual Learning

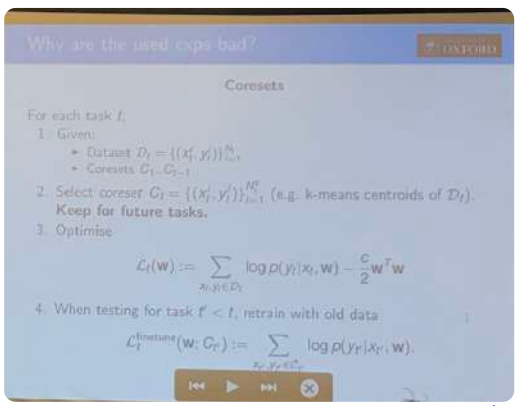
- Continual learning ⇒ streaming of data → adapt to new batches of data
  - ↳ E.g. dynamic models, discarding data after learning

Need:

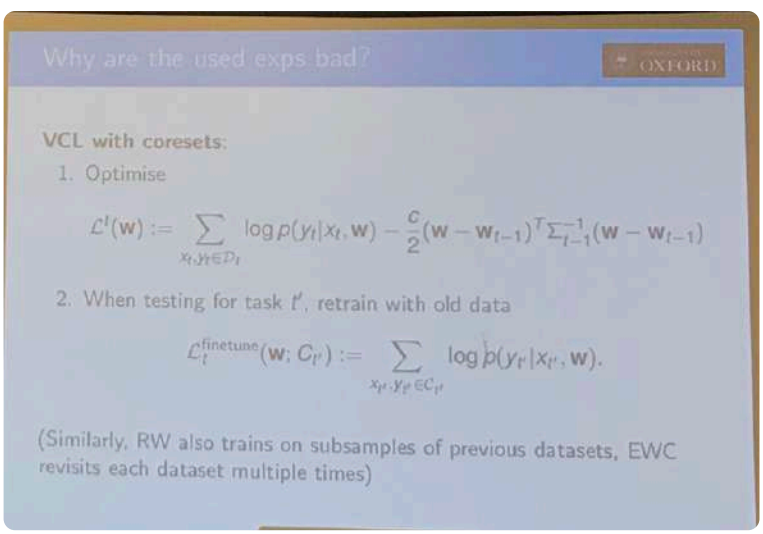
- Cross-talk resemblance → Multi-headed split MNIST → 0 vs 1 → 2 vs 3 (multi-class add. info!)
- Shared output head → no catastrophic forgetting
- No re-training on old tasks
- Have them 2 tasks
- Don't know task id at test time → which task are we trying to answer at test
- Real time updates // small memory footprint



Posterior Passing Tasks



Coresets Previous Tasks



Combine both ideas!

Permuted vs. Split

Permuted: no cross-task resemblance

Split: yes cross-task resemblance

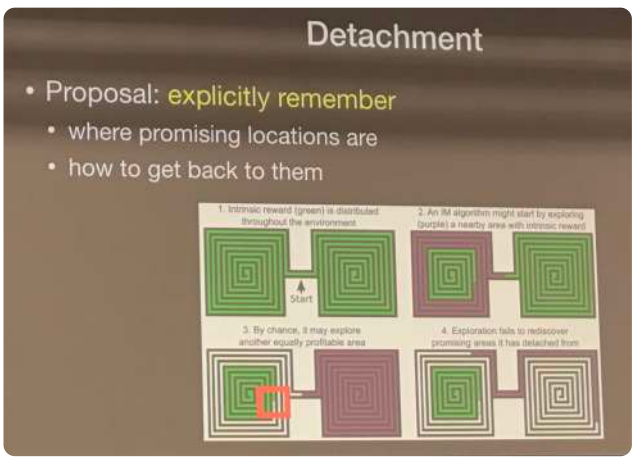
- Work on style-headed MNIST → hard separate > hard datasets
- Discuss when task has changed!

⑨ J. Clune – Go-Explore → Algo for Hard-Exploration Problems

- Hard Exploration Problems → sparse-rewards ⇒ not
  → deceptive problems ⇒ many → local optima
- Intrinsic Motivation → Boredom based (Curiosity) ⇒ Montezuma's Surge

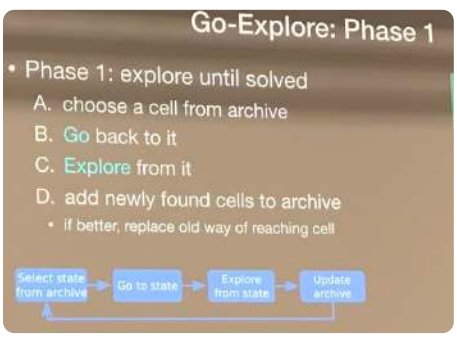
↘↗ backtracking after no intrinsic motivation left → "Detachment"
↳ not solved by buffer

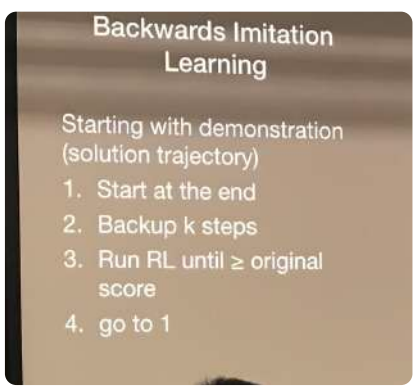
- "Derailment"
  ↳ first return, then explore!
  ↳ robustness effects?
  ⇒ first solve task, then robustify ① ②



→ all representations ⇒ similarity after down-scaling

↳ Phase 2: Robustify ⇒ add noise!
via backwards imitation learning
⇒ DISCUSSION STOCHASTICITY TRAIN/TEST
↳ 2 modes of backwards!





(10) Bayesian DL Panel

Amortized inference posit frequentist