Beyond individual algorithms: Computational architectures for reinforcement learning in AI

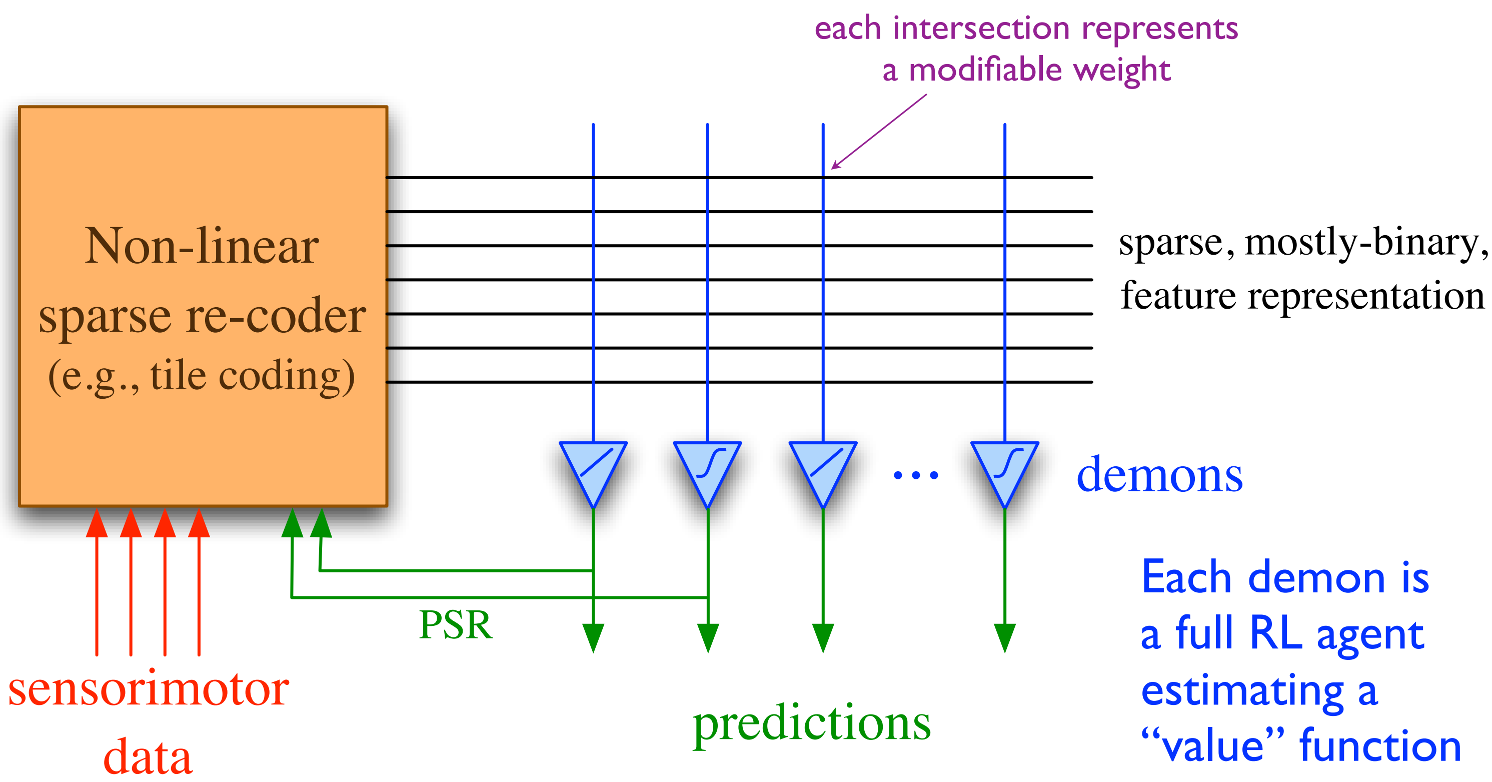
Doina Precup
McGill University
With thanks to Rich Sutton

Barbados Reinforcement Learning Workshop, 2015

Taking stock of current RL state-of-art

- A lot of successful work on algorithms for solving specific problems
  - Temporal-difference learning
  - Convergent methods of off-policy learning
  - Eligibility traces
  - Policy gradient methods and other types of policy search
  - Learning and planning with temporally extended actions
  - Exploration methods (though here there is much to do still)
  - Sample-based planning (eg. Monte Carlo Tree search)
- As in the rest of AI, progress was made by breaking off manageable pieces and working on them
- It is now time to think again about how the pieces can be put together in order to form an RL architecture

Example architecture: Horde

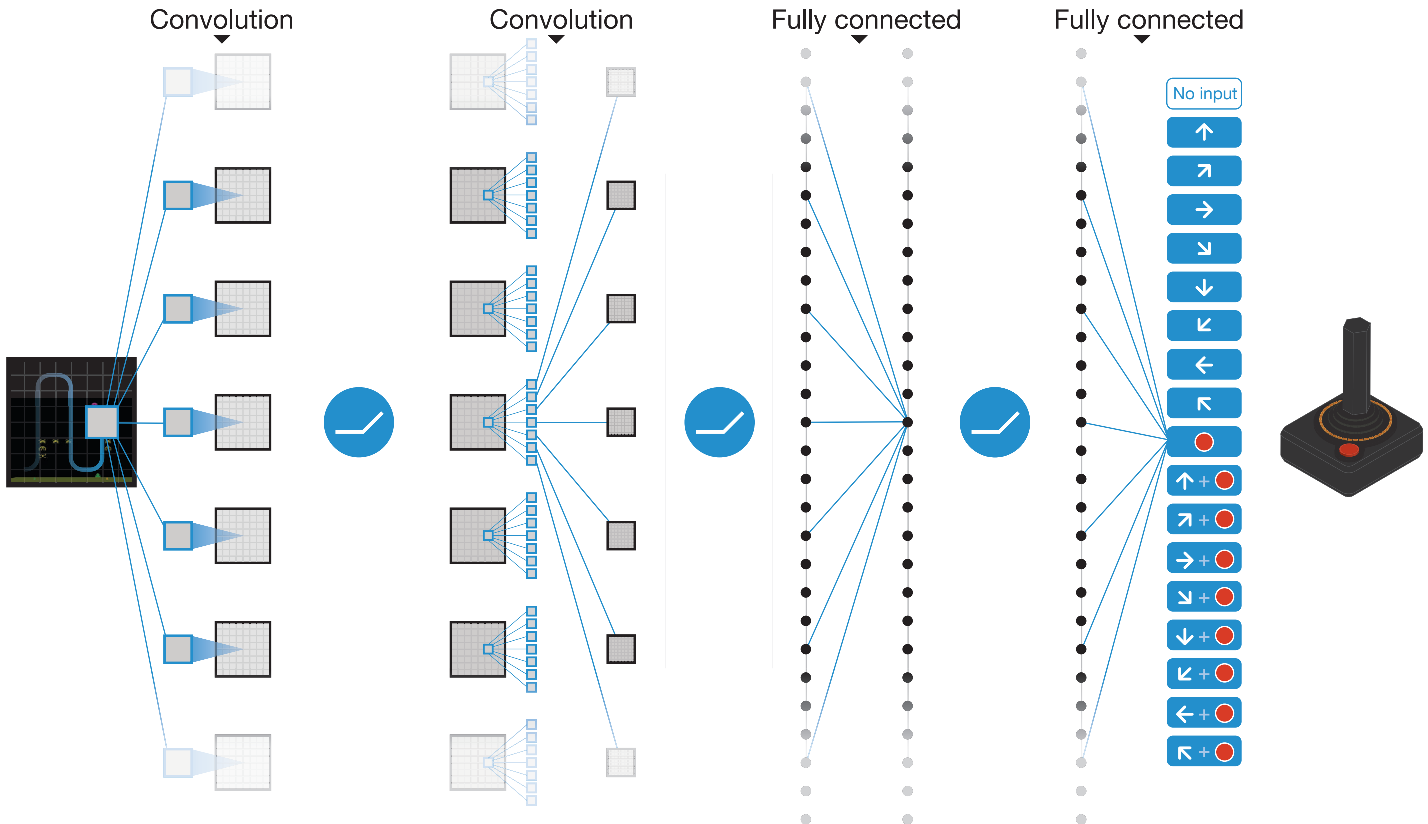


Cf. Sutton et al, AAMAS 2011

Horde: Characteristics

- Many value functions for different time scales and reward functions are learned in parallel, off-policy, from one stream of experience
- Emphasis on function approximation and incremental learning
- Exploration is still an open problem
- Several decision-making mechanisms possible
  - Winner-take-all
  - Separate "driving" value function
  - Some kind of committee-based decision?
- Planning in the horde has been less explored than learning

Example architecture: DQN (Google Deepmind)



Cf. Human-level control through deep reinforcement learning, Mnih, Kavukcuoglu, Silver et al, Nature, 2015

DQN: Characteristics

- Relies on deep neural networks as very powerful feature extractors
- Experience replay is used to allow stable training of the deep nets
  - Sample transitions picked in random order not in sequence
  - Parameters for computing the target values are not changed after every update
- Exploration done as usual
- No temporal abstraction or planning in this published version

Desiderata for an RL architecture

- “Fast” acting process: has to respond quickly to the environment
- “Slow” planning process: deliberation takes time, informs the low level
- Predictions about future courses of action should be useful:
  - as features
  - for planning

  Cf. Horde, predictive state representations, recurrent networks
- Stable learning should happen both for the “fast” and “slow” level
- Off-policy learning and eligibility traces are necessary

Acting

- Having a separate actor seems important in order to achieve fast acting
- This is especially true in continuous/large action spaces, where computing max might be infeasible
- Therefore, *actor-critic* seems the obvious choice of architecture
- Actor should take as input current features as well as information from the planner and produce an action
- What is the right way to obtain the actor? Policy gradient over options or primitive actions?

Planning

- Need *models that make predictions about the effects of actions*, i.e. predictive state representations
- *Models that are compositional*, can be used to reason about sequences of actions
- Eg., if we want Bellman equations to hold, we need:

$$\begin{aligned} \mathbf{r}_{o_1 o_2} &= \mathbf{r}_{o_1} + \mathbf{p}_{o_1} \mathbf{r}_{o_2} \\ \mathbf{p}_{o_1 o_2} &= \mathbf{p}_{o_1} \mathbf{p}_{o_2} \end{aligned}$$

- The demons in Horde are a form of model, although not trained specifically to be compositional
- Dyna using models should be very useful

Formalizing the higher-level goals of the agent

- *Intentions*: reward functions that the agent is trying to maximize
- Helpful to think of them as subgoals that the agent might try to achieve
- Several intentions can be active at the same time
- One can think of intentions as defining the space of possible options at a given point in time
- The planner communicates the current set of active intentions to the actor, which uses them as part of its internal state

More on intentions

- Intentions give rise to policies, so they are close to parameterized options
- Models need to predict the future given the current set of intentions and current observations (and perhaps current predictions about the future)
- Ultimately, we want to be able to generate *gradients* for the actor as well as the models / value functions, so compositionally may not be a stringent requirement anymore
- The way intentions are expressed can be used to control complexity in the system

Learning

- A rich set of ideas has been developed in the last few years on how to achieve off-policy learning in an efficient manner
  - Gradient-based TD
  - Dutch traces
  - Emphasis TD
- Importance sampling is still a key ingredient, and potential source of high variance
- Idea: Learn approximations to the importance sampling ratios, instead of insisting that the ratios are computed based on known behaviour and target policies

Questions for the workshop

- What are your wishes for an RL architecture?
- Are the architectures we have already sufficient to achieve or goals of RL-based AI?
- What basic algorithmic pieces do we still need to build?
  - Exploration seems to be still a big open issue
- What is the best way to interleave planning, acting and learning