ConvNets and Forward Modeling for StarCraft AI

Alex Auvolat

September 15, 2016

Overview

Section 1

ConvNets for StarCraft

A common architecture for forward modeling and RL

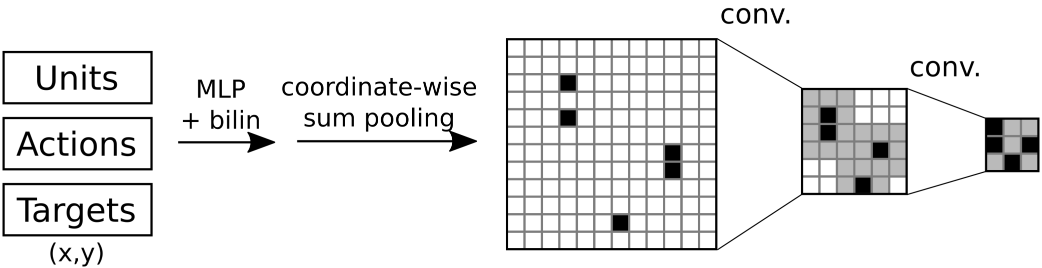
The idea:

- Network input = 2D "image" of game state
- 1 ConvNet pixel = 1 game walktile

Why ConvNets:

- Natural representation
- Implicit encoding of relative positions
- Possibility of handling collisions
- Possibility of handling complex actions with area of effect (e.g. Psi Storm)

Network structure



Example

Two ally units at $(4, 3)$ and $(10, 7)$ attacking a single enemy unit at $(4, 5)$

(x, y)	**Type**	**Meaning**
$(4, 3)$	Unit	Ally Terran Marine present here, 40 HP
$(4, 3)$	Action	Ally Terran Marine here attacking at $(+0, +2)$, 0 cooldown
$(10, 7)$	Unit	Ally Terran Marine present here, 12 HP
$(10, 7)$	Action	Ally Terran Marine here attacking at $(-6, -2)$, 5 frames cooldown
$(4, 5)$	Unit	Enemy Terran Marine present here, 25 HP
$(4, 5)$	Target	Ally Terran Marine attacking here from $(+0, -2)$, 0 cooldown
$(4, 5)$	Target	Ally Terran Marine attacking here from $(+6, +2)$, 5 frames cooldown

Table: Feature vectors for a simple example state

Section 2

Forward Modeling

Real Game Example

[VIDEO]

StarCraft ConvNet for Forward Modeling

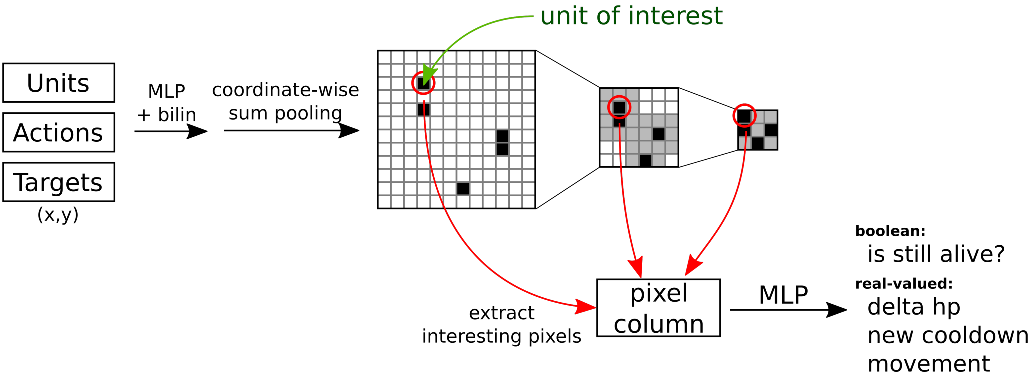
Method:

- Extract pixel of a unit $\rightarrow$ MLP $\rightarrow$ predict unit's next state
- Use human player data as training set
- Predict game state at $t + 8$ frames

Possible Uses:

- Tree search
- Share parameters with RL model, learn better features for transfer learning
- Instead of evaluating $Q(s, a)$, calculate estimation of state s' and evaluate $V(s')$
- Model-based RL

Network structure



Experiment details

Data set: 7000 pro human games (176 789 battles, $>$ 100 frames each)

- Train set $=$ 172591 battles
- Test set $=$ 2153 battles (from different games)
- 110 unit types, 180 action types

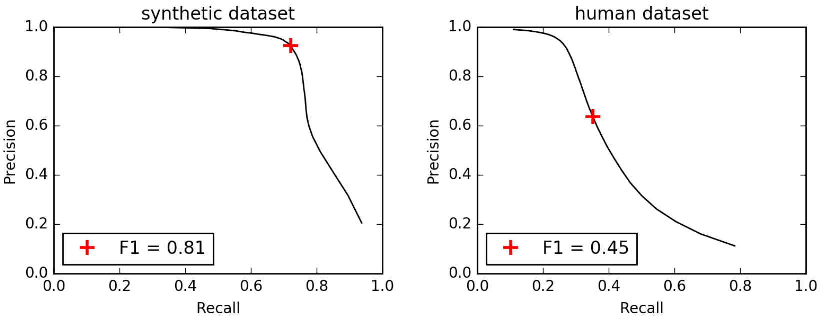
Evaluation:

- Synthetic dataset, same small scenarios as in RL task
- Human dataset

Baseline:

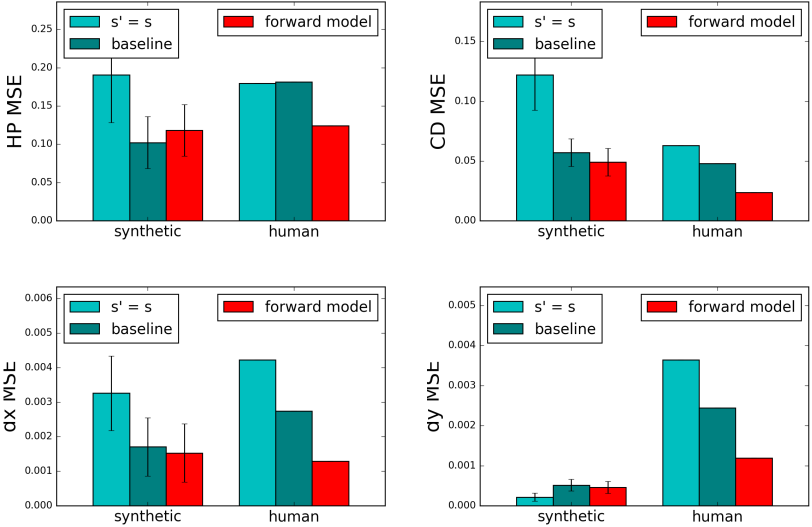
- Hand-crafted approximation of the game dynamics: dealing with attacks and movements, rules for velocity and acceleration.
- Lacks many corner cases. No handling of collisions, . . .

Results: precision/recall on dead unit prediction



	Synthetic dataset			Human dataset		
	Precision	Recall	F1	Precision	Recall	F1
Baseline	0.12	0.12	0.12	0.07	0.02	0.03
Forward model	**0.92**	**0.72**	**0.81**	**0.63**	**0.35**	**0.45**

Results: mean square errors



Analysis

Results:

- Forward model works much better than hand-crafted heuristic
- Particularly clear on dead/alive prediction

Conclusion:

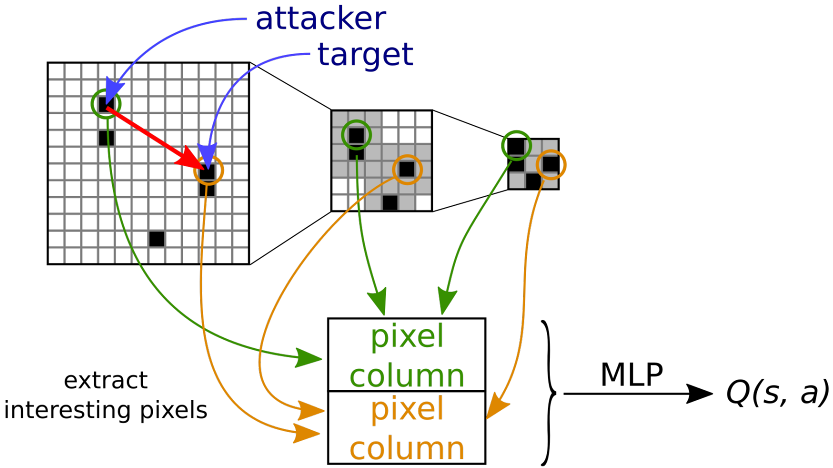
- StarCraft dynamics are complex, difficult to approximate with a small set of rules
- $\rightarrow$ Need a model that can learn from examples!
- Still room for model improvements (e.g. buildings)

Section 3

Reinforcement Learning with ConvNets

Example scenario

Network structure



Where we're at

What is coded:

- RL model from scratch
- RL model with transfer learning (taking parameters from the forward model)
- Parameter freeze vs. parameter fine-tuning

Preliminary results:

- Transfer learning might help on `m5v5`, still running
- Pre-training has not yet enabled us to train a ConvNet model on bigger maps such as `m15v16`

Conclusion

Status:

- The forward model on its own beats a reasonably good baseline, showing that learning is useful
- RL experiments in progress

Other ideas:

- Tree search
- Imitation learning
- Structure learning

Questions?