Universal Value Function Approximators

Tom Schaul, Dan Horgan, Karol Gregor, Dave Silver
Motivation

**Forecasts** about the environment

- = temporally abstract predictions (questions)
Motivation

**Forecasts** about the environment

- = temporally abstract predictions (questions)
- not necessarily related to reward
- conditioned on a behavior
- (aka GVF, nexting)
Motivation

Forecasts about the environment

• = temporally abstract predictions (questions)
• not necessarily related to reward
• conditioned on a behavior
• (aka GVFs, nexting)
• many of them
Motivation

Forecasts about the environment

- = temporally abstract predictions (questions)
- not necessarily related to reward
- conditioned on a behavior
- (aka GVF, nexting)
- many of them

Why?

- better, richer representations (features)
- decomposition, modularity
- temporally abstract planning, long horizons
Concretely

Subgoal forecasts
Concretely

Subgoal forecasts

• Reaching any of a set of states, then
  • the episode terminates ($\gamma = 0$)
  • and a pseudo-reward of 1 is given
Concretely

Subgoal forecasts

- Reaching any of a set of states, then
  - The episode terminates ($\gamma = 0$)
  - And a pseudo-reward of 1 is given
- Various time-horizons induced by $\gamma$
Concretely

Subgoal forecasts
• Reaching any of a set of states, then
  • the episode terminates ($\gamma = 0$)
  • and a pseudo-reward of 1 is given
• Various time-horizons induced by $\gamma$

Q-values
Concretely

Subgoal forecasts

• Reaching any of a set of states, then
  • the episode terminates ($\gamma = 0$)
  • and a pseudo-reward of 1 is given

• Various time-horizons induced by $\gamma$

Q-values

• for the subgoal-optimal policy
Concretely

Subgoal forecasts
• Reaching any of a set of states, then
  • the episode terminates ($\gamma = 0$)
  • and a pseudo-reward of 1 is given
• Various time-horizons induced by $\gamma$

Q-values
• for the subgoal-optimal policy

Neural networks as function approximators
Combinatorial numbers of subgoals
Combinatorial numbers of subgoals

Why?

- because the environment admits tons of forecasts
- any of them could be useful for the task
Combinatorial numbers of subgoals

Why?
• because the environment admits tons of forecasts
• any of them could be useful for the task

How?
Combinatorial numbers of subgoals

Why?
• because the environment admits tons of forecasts
• any of them could be useful for the task

How?
• efficiency
Combinatorial numbers of subgoals

Why?
• because the environment admits tons of forecasts
• any of them could be useful for the task

How?
• efficiency
• exploit shared structure in value space
Combinatorial numbers of subgoals

Why?
• because the environment admits tons of forecasts
• any of them could be useful for the task

How?
• efficiency
• exploit shared structure in value space
Combinatorial numbers of subgoals

Why?
• because the environment admits tons of forecasts
• any of them could be useful for the task

How?
• efficiency
• exploit shared structure in value space
• generalize to similar subgoals
Outline

• Motivation
  • learn values for forecasts
  • efficiently for many subgoals

• Approach
  • new architecture
  • one neat trick

• Results
Universal Value Function Approximator

- a single neural network producing $Q(s, a; g)$
Universal Value Function Approximator

- a single neural network producing $Q(s, a; g)$
  - for many subgoals $g$
  - generalize between subgoals
  - compact
Universal Value Function Approximator

- a single neural network producing $Q(s, a; g)$
  - for many subgoals $g$
  - generalize between subgoals
  - compact

- UVFA ("you-fah")
UVFA architectures

- Vanilla (monolithic)
UVFA architectures

- Vanilla (monolithic)
- Two-stream
UVFA architectures

- Vanilla (monolithic)
- Two-stream
  - separate embeddings $\phi$ and $\psi$ for states and subgoals
  - Q-values = dot-product of embeddings
UVFA architectures

- Vanilla (monolithic)
- Two-stream
  - separate embeddings $\varphi$ and $\psi$ for states and subgoals
  - Q-values = dot-product of embeddings
  - (works better)
UVFA learning

- Method 1: bootstrapping

\[ Q(s_t, a_t, g) \leftarrow \alpha \left( r_g + \gamma_g \max_{a'} Q(s_{t+1}, a', g) \right) + (1 - \alpha) Q(s_t, a_t, g) \]
UVFA learning

- Method 1: bootstrapping

\[
Q(s_t, a_t, g) \leftarrow \alpha \left( r_g + \gamma_g \max_{a'} Q(s_{t+1}, a', g) \right) \\
+ (1 - \alpha) Q(s_t, a_t, g)
\]

- some stability issues
UVFA learning

- **Method 1: bootstrapping**
  \[
  Q(s_t, a_t, g) \leftarrow \alpha \left( r_g + \gamma_g \max_{a'} Q(s_{t+1}, a', g) \right) \\
  + (1 - \alpha) Q(s_t, a_t, g)
  \]
  
  - some stability issues

- **Method 2:**
  - built training set of subgoal values
  - train with supervised objective
  - like neuro-fitted Q-learning
UVFA learning

• **Method 1:** bootstrapping

\[
Q(s_t, a_t, g) \leftarrow \alpha \left( r_g + \gamma_g \max_{a'} Q(s_{t+1}, a', g) \right) \\
+ (1 - \alpha) Q(s_t, a_t, g)
\]

• some stability issues

• **Method 2:**
  • built training set of subgoal values
  • train with supervised objective
  • like neuro-fitted Q-learning
  • (works better)
Outline

• Motivation
  • learn values for forecasts
  • efficiently for many subgoals

• Approach
  • new architecture: UVFA
  • one neat trick

• Results
Trick for supervised UVFA learning: FLE
Trick for supervised UVFA learning: FLE

Stage 1: Factorize

\[ Q(s,a;g) \]

\[ \phi(s,a) \] \[ \psi(g) \]
Trick for supervised UVFA learning: FLE

Stage 1: **Factorize**
Stage 2: **Learn Embeddings**
Stage 1: Factorize (low-rank)
Stage 1: Factorize (low-rank)

- target embeddings for states and goals
Stage 2: Learn Embeddings

- regression from state/subgoal features to target embeddings
Stage 2: Learn Embeddings

- regression from state/subgoal features to target embeddings

(optional Stage 3): end-to-end fine-tuning
FLE vs end-to-end regression

- between 10x and 100x faster
Outline

- Motivation
  - learn values for forecasts
  - efficiently for many subgoals
- Approach
  - new architecture: UVFA
  - one neat trick: FLE
- Results
Results: Low-rank is enough
Results: Low-rank embeddings
Results: Low-rank embeddings
Results: Generalizing to new subgoals
Results: Transfer to new subgoals

Refining UVFA is much faster than learning from scratch
Results: Pacman pellet subgoals

training set

[Image of Pacman game maze]

test set

[Image of Pacman game maze]
Results: pellet subgoal values (test set)

“truth”
Results: pellet subgoal values (test set)

“truth”

UVFA generalization
Summary

- UVFA
  - compactly represent values
  - for many subgoals
  - generalization
  - transfer learning
Summary

• **UVFA**
  • compactly represent values
  • for many subgoals
  • generalization
  • transfer learning

• **FLE**
  • a trick for efficiently training UVFAs in 2 stages
  • side-effect: interesting embedding spaces
  • scales to complex domains (Pacman from raw vision)
Bonus results: Extrapolation

even to subgoals in unseen fourth room:

truth

UVFA
Bonus results: Extrapolation

even to subgoals in unseen fourth room:

truth

UVFA