1. **Options**: Use “exit this room” high-level action

2. **Hierarchical Reinforcement Learning:** Attack, Mine Gold, Explore subtasks

3. **Learning from Demonstration**: Control Mario and have him learn to mimic you

4. **Ensembles**: Combine multiple classifiers to improve accuracy

5. **Unsupervised Learning:** Which players have similar playing styles?

6. **Active Learning:** Program asks which points should be labeled

7. **Other**?

- Monday: Project idea
- What to cover next?

Combination Techniques

1. $R'(s,a) = R(s,a) + (weight * \hat{H}(s,a))$.
2. $\overrightarrow{f'} = \overrightarrow{f}.append(\hat{H}(s,a))$.
3. *Initially train* $Q(s,a)$ *to approximate* $(constant * \hat{H}(s,a))$.
4. $Q'(s,a) = Q(s,a) + constant * \hat{H}(s,a)$.
5. $A' = A \cup argmax_a[\hat{H}(s,a)]$.
6. $a = argmax_a[Q(s,a) + weight * \hat{H}(s,a)]$.
7. $P(a = argmax_a[\hat{H}(s,a)]) = p$. *Otherwise original RL agent's action selection mechanism is used.*
8. $R'(s_t,a) = R(s,a) + constant * (\phi(s_t) - \phi(s_{t-1})$, *where* $\phi(s) = max_a H(s,a)$.

Combination Techniques

1. $R'(s,a) = R(s,a) + (weight * \hat{H}(s,a))$.
2. $\overrightarrow{f'} = \overrightarrow{f}.append(\hat{H}(s,a))$.
3. *Initially train* $Q(s,a)$ *to approximate* $(constant * \hat{H}(s,a))$.
4. $Q'(s,a) = Q(s,a) + constant * \hat{H}(s,a)$.
5. $A' = A \cup argmax_a[\hat{H}(s,a)]$.
6. $a = argmax_a[Q(s,a) + weight * \hat{H}(s,a)]$.
7. $P(a = argmax_a[\hat{H}(s,a)]) = p$. *Otherwise original RL agent's action selection mechanism is used.*
8. $R'(s_t,a) = R(s,a) + constant * (\phi(s_t) - \phi(s_{t-1})$, *where* $\phi(s) = max_a H(s,a)$.

reward shaping

Combination Techniques

1. $R'(s,a) = R(s,a) + (weight * \hat{H}(s,a))$.
2. $\vec{f'} = \vec{f}.append(\hat{H}(s,a))$.
3. Initially, $Q(s,a)$ is approximated $(constant * \hat{H}(s,a))$.
4. $Q'(s,a) = Q(s,a) + constant * \hat{H}(s,a)$.
5. $A' = A \cup argmax_a[\hat{H}(s,a)]$.
6. $a = argmax_a[Q(s,a) + weight * \hat{H}(s,a)]$.
7. $P(a = argmax_a[\hat{H}(s,a)]) = p$. *Otherwise original RL agent's action selection mechanism is used.*
8. $R'(s_t,a) = R(s,a) + constant * (\phi(s_t) - \phi(s_{t-1})$, *where* $\phi(s) = max_a H(s,a)$.

Q augmentation

Combination Techniques

1. $R'(s,a) = R(s,a) + (weight * \hat{H}(s,a))$.
2. $\overrightarrow{f'} = \overrightarrow{f}.append(\hat{H}(s,a))$.
3. *Initially train* $Q(s,a)$ *to approximate* $(constant * \hat{H}(s,a))$.
4. $Q'(s,a) = Q(s,a) + constant * H(s,a)$.
5. $A' = A \cup argmax_a[\hat{H}(s,a)]$.
6. $a = argmax_a[Q(s,a) + weight * \hat{H}(s,a)]$.
7. $P(a = argmax_a[\hat{H}(s,a)]) = p$. *Otherwise original RL agent's action selection mechanism is used.*
8. $R'(s_t,a) = R(s,a) + constant * (\phi(s_t) - \phi(s_{t-1})$, *where* $\phi(s) = max_a H(s,a)$.

action biasing

Combination Techniques

1. $R'(s,a) = R(s,a) + (weight * \hat{H}(s,a))$.
2. $\overrightarrow{f'} = \overrightarrow{f}.append(\hat{H}(s,a))$.
3. *Initially train* $Q(s,a)$ *to approximate* $(constant * \hat{H}(s,a))$.
4. $Q'(s,a) = Q(s,a) + constant * \hat{H}(s,a)$.
5. $A' = A \cup argmax_a[\hat{H}(s,a)]$.
6. $a = argmax_a[Q(s,a) + weight * \hat{H}(s,a)]$.
7. $P(a = argmax_a[\hat{H}(s,a)]) = p$. *Otherwise original RL agent's action selection mechanism is used.*
8. $R'(s_t,a) = R(s,a) + constant * (\phi(s_t) - \phi(s_{t-1})$, *where* $\phi(s) = max_a H(s,a)$.

control sharing

Experiments

- domain: Mountain Car
- RL algorithm: Sarsa(λ)
  - features: a grid of 2D Gaussian RBFs over state; one grid for each action
  - representation of Q: linear model
  - initialization of Q: both opt. and pess.
  - updates: gradient descent
- 30 runs of 500 episodes

Experiments

Two predictive models used (from among 19 trainers):

$\hat{H}_1$: middling performance (9th)

$\hat{H}_2$: best performance

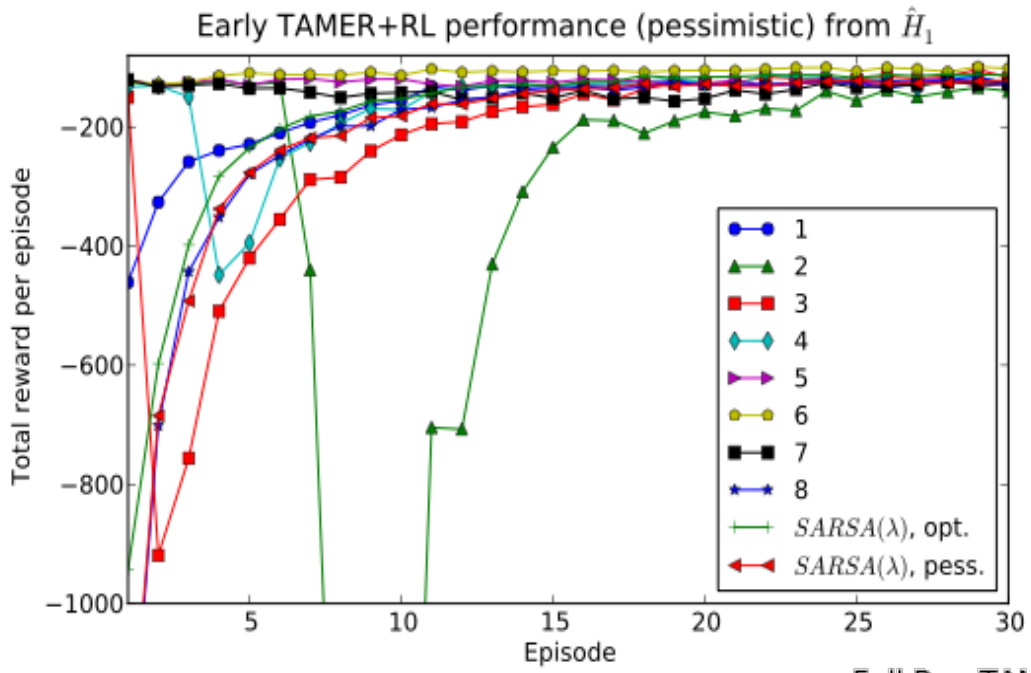
Defining success

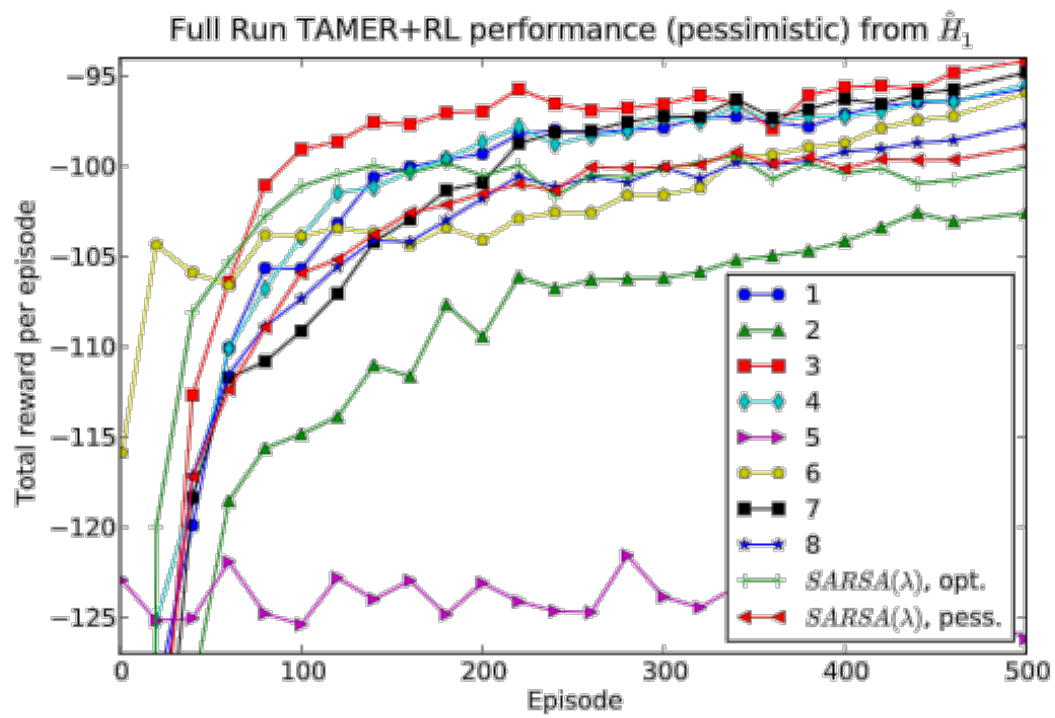
Outperforming:

On the metrics:

	TAMER-only	RL-only
cumulative reward	?	?
final performance	?	?

On both $\hat{H}_1$ and $\hat{H}_2$





(for comparison, TAMER-only mean performance is -109.1)

Almost complete successes

$$R'(s,a) = R(s,a) + (weight * \hat{H}(s,a))$$ **reward shaping**

and

$$Q'(s,a) = Q(s,a) + constant * \hat{H}(s,a)$$ **Q augmentation**

Outperforming:

On the metrics:

	TAMER-only	RL-only
cumulative reward	only on $\hat{H}_1$	
final performance		

Complete successes

$$a = argmax_a[Q(s,a) + weight * \hat{H}(s,a)]$$

action biasing

and

$$P(a = argmax_a[\hat{H}(s,a)]) = p. \ Otherwise \ original \ RL \ agent's \ action \ selection \ mechanism \ is \ used.$$

control sharing

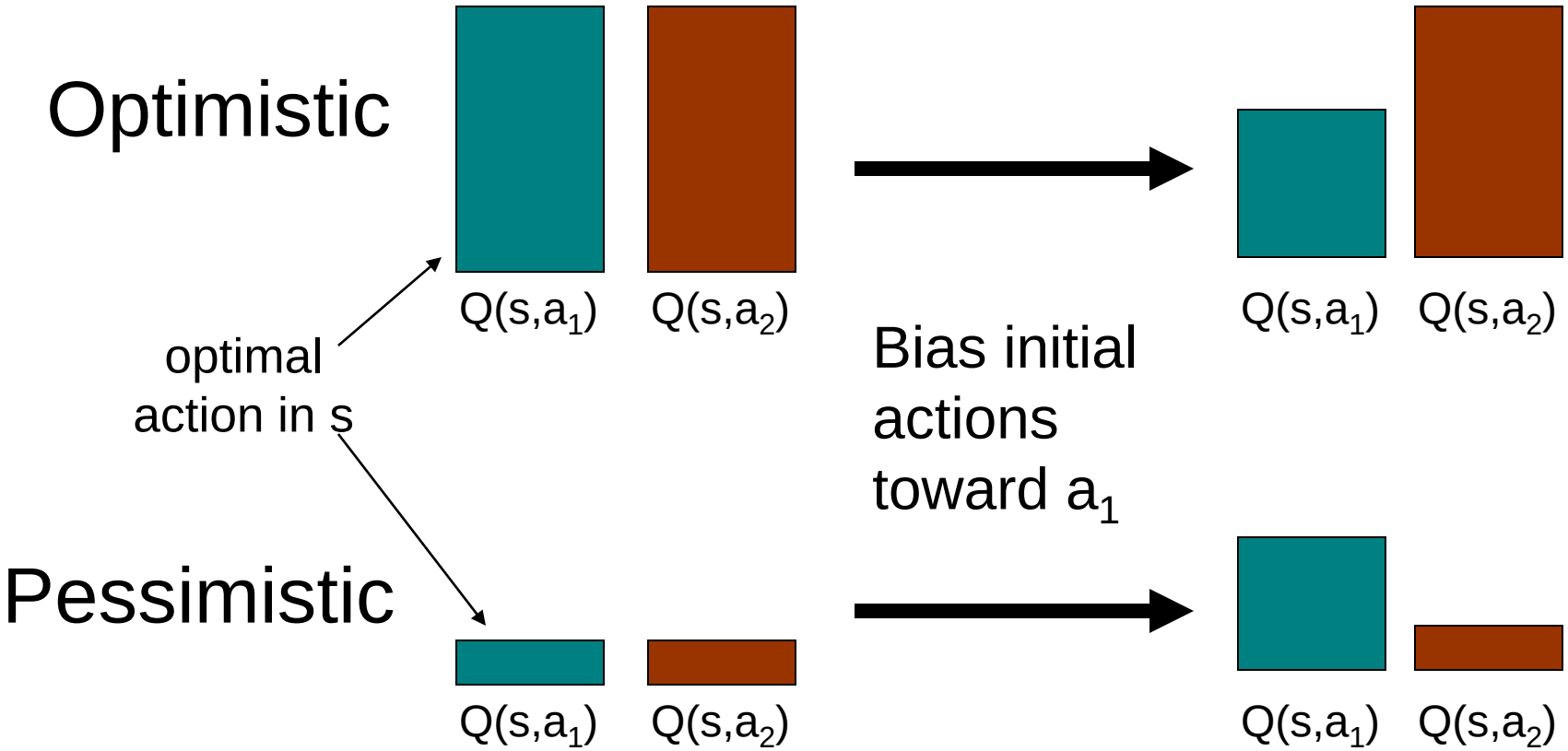
Outperforming:

On the metrics:

	TAMER-only	RL-only
cumulative reward		
final performance		

Lessons

1. Optimistic vs. pessimistic initialization



Lessons

2. Biasing action selection was most effective

- better than shaping rewards

Summary

- background: TAMER Framework for learning from interactive shaping
- explored ways of combining TAMER with RL
- identified several successful techniques and some general lessons

- future: more domains, more RL algorithms