Curriculum Learning via Reward Shaping

Jackson de Campos, Aidan Cassel-Mace, Eliot Laidlaw, Adam Pikielny, Kshitij Sachan
PMAAPOM

Final Project for CSCI2951f: Sequential Decision Making

December 18, 2021

Abstract: Recent work has shown that learning on complex problems can be sped up by first learning a series of simpler source tasks. That knowledge can then be transferred via reward shaping to tackle the more challenging original problem. One limitation to this approach is that the source and target tasks generally need to have similar state/action spaces, and choosing good source tasks often requires careful testing. We explore these limitations with gridworld experiments and introduce the idea of mapping state-action pairs in a target task to corresponding state-action pairs in the source tasks, regardless of whether the tasks share the same exact state or action spaces. We demonstrate how this more flexible state-action mapping can be used to solve the pursuit domain. We also explore different techniques to aggregate information from multiple source tasks.

1 Introduction

Humans learn complex tasks by breaking them into simpler ones. Before we start driving on highways, we practice in a parking lot. Before riding a bike, we might practice on a tricycle. In contrast, most reinforcement learning agents attempt to solve an entire task with random actions and then hone their strategy through trial and error. This is especially true when the agent only receives sparse rewards, which is a type of reward function where positive reward is only given in a few states.

Learning from sparse reward is important because a sparse reward is often the most obvious way to define success in a task. For example, the only objective in many games like chess is to win, so a natural reward function is -1 for a loss, +1 for a win, and 0 reward at every other step. Learning the optimal strategy in a game as complex as chess with such a sparse reward function is extremely challenging.

Our project attempts to mitigate the problem of sparse rewards by learning a curriculum of tasks with increasing complexity, building up to the target task (such as learning to ride a tricycle before a bike). The source tasks should be simple enough to be learned with a sparse reward. This approach is known as curriculum learning.

We explore using reward shaping—a specific way to transfer knowledge between tasks—with curriculum learning to speed up learning. We specifically focus on how to design a good curriculum and how to learn with multiple source tasks.

2 Background / Related Work

2.1 Q-Learning

Q-learning is a model-free reinforcement learning algorithm that estimates the value function, $Q(s, a)$, which represents the expected reward after taking action a from state s. At each step, the Q function is updated using the following formula: [1].

$$Q'(s, a) = Q(a, a) + \alpha(r + \gamma \text{max}_{a'} Q(s', a') - Q(s, a))$$

2.2 Reward Shaping

Reward shaping is a technique to make learning easier in sparse reward problems by providing supplementary reward [2]. For example, the point values of chess pieces serve as a proxy reward function for humans learning to play chess. However, reward shaping does not guarantee the optimal policy in the original MDP. Many beginner chess players, for example, overlook strategies that sacrifice a piece (and hence points) for some other kind of tactical advantage because these strategies are not maximizing the points reward function. There are other examples in the literature of reward hacking, where an agent loops behavior to receive a shaped reward bonus that does not correspond to progress in the unshaped MDP, such as a robot vacuum cleaner ejecting dust to sweep it up again [5].

Potential-based reward shaping is a specific way to provide shaped reward that does not affect the optimal policy [4]. In its simplest formulation, each state in the MDP is provided a certain value or potential, $\phi(s)$. The shaped reward for transitioning from state s to s' is $F(s, a, s') = \gamma\phi(s') - \phi(s)$, where γ is the discount factor. This potential-based framework prevents self-loops from incurring a positive reward because $F(s_1, a_1, s_2) + \ldots + F(s_{n-1}, a_{n-1}, s_n) + F(s_n, a_n, s_1) = 0$. If the reward in the original MDP was provided by $R(s, a, s')$, then the reward function in the shaped MDP is $R'(s, a, s') = F(s, a, s') + R(s, a, s')$.

Although potential-based reward shaping doesn't change the optimal policy in the limit, it does change the amount of time required to learn the optimal policy. The shaping function that leads to the quickest learning is using the optimal Q values as the potential function[4].

2.3 Curriculum learning

Curriculum learning involves training the agent on simpler source tasks and then using the learned knowledge in a harder target task [3]. The curriculum is a directed acyclic graph, where nodes are tasks and an edge from one task to another indicates that the source task must be learned before the target task. A major complication with curriculum learning is specifying how to actually transfer knowledge between tasks. Models, value functions, training examples, policies, and even partial policies (options) can be transferred.

2.4 Reward Shaping + Curriculum learning

Svetlik et al. [6] used reward shaping as a transfer method for curriculum learning. The value function in the source task was used as the potential function for reward shaping in the target task. To understand exactly how reward shaping is applied, consider two MDPs: the source and target tasks. These MDPs may have different state spaces, S_{source} and S_{target}. We can partition the vector representing a state in the target task into its components in the source task's state space and not in the source task's state space: $s_{target} = [s_{source}, s_{target \setminus source}]$. The potential shaping function for the target task is defined as:

$$\phi(s_{target}, a) = \begin{cases} Q_{source}(s_{source}, a) & s_{source} \in S_{source} \\ 0 & \text{else} \end{cases}$$

Therefore, a shaped reward is only provided if the current state can be found in the source MDP. This is a brittle transfer approach (see 1) that works in carefully crafted cases but has major limitations. Breaking the state into components does not work when the action spaces of the source and target tasks are not the same or when the state spaces are not sufficiently similar. In section 4.2, we will define our more robust transfer approach.

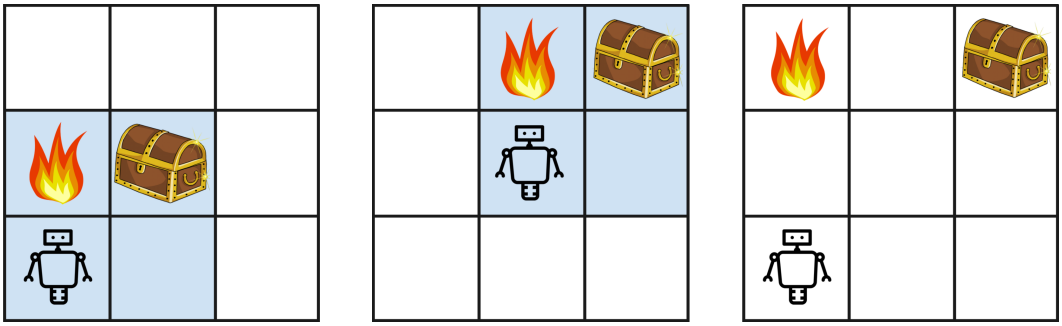


Figure 1: Curriculum learning in a Gridworld domain. The agent must navigate to the treasure while avoiding the fire. The state is the relative position of the fire and treasure to the agent. Left: The source task. Middle: A target task. The shaded squares are the possible agent positions that would receive a reward shaping bonus (where the bonus would come from the agent positions in the shaded square in the target task on the left). For example, the current state is fire=(0,1) and treasure=(1,1) with respect to the agent, and this exactly matches the state we see in the left target task. Right: A different target task where no transfer can occur, regardless of the agent's position, because there is a gap between the fire and treasure that doesn't exist in the source task. Designing an entire curriculum while preventing this case from occurring is challenging.

One caveat is that there can be more than one source task for a target task because, as mentioned above, a curriculum is generally a DAG, not a sequence. To combine knowledge from more than one source task, Svetlik et al. [6] added the reward shaping potential from all source tasks when solving the target task. In section 4.2, we explore the problem of combining information from multiple source tasks and explore different aggregation functions besides addition.

3 Problem Statement

Given a target MDP parameterized as (S_t, A_t, T_t, R_t), we attempt the following tasks:

1. Define one or more source MDPs, parameterized as $\{(S_{s_1}, A_{s_1}, T_{s_1}, R_{s_1}), ...\}$ such that when a Q-learning agent has learned an optimal policy on the source task(s), there will be valuable information to transfer to a Q-learning agent on the target task.
2. Given the Q functions from optimized agents on the source tasks, $\boldsymbol{Q_s} = \{Q_{s_1}, ..., Q_{s_n}\}$, define a potential-based reward shaping function $\phi(s_t, a_t)$ that draws from the Q values in $\boldsymbol{Q_s}$, such that when this function is used to shape reward for a Q learner on the target task, the Q learner learns sufficiently faster or sufficiently than a randomly initialized Q learner to justify the time taken to learn $\boldsymbol{Q_s}$.

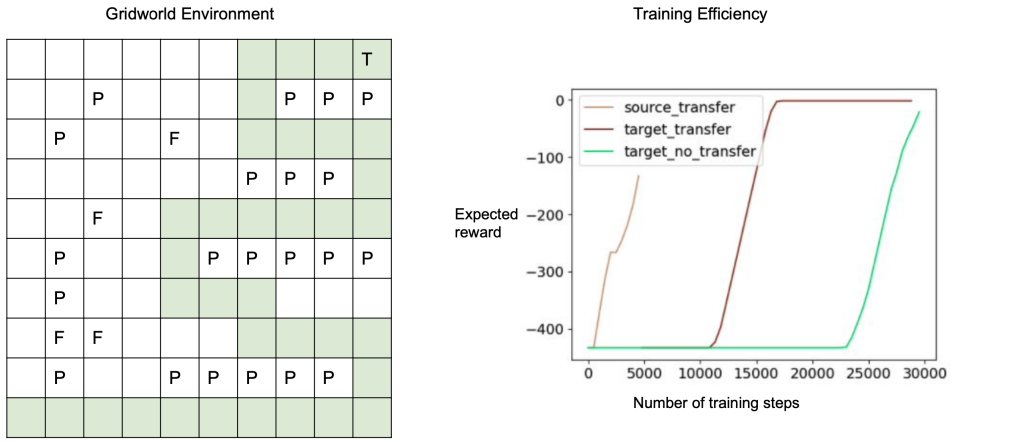


Figure 2: Left: a representation of the maze gridworld where cells containing "F" contain fires, those containing "P" contain pits, and "T" represents the treasure. The agent begins in the bottom left corner, and the optimal path is highlighted in green. Right: The learning curve with transfer learning.

4 Methodology and Experiments

4.1 Experiment: Transfer learning in Gridworld

First, we looked at a simple gridworld domain proposed by Svetlik et al. [6] to show that transfer learning could improve the speed of Q-learning. The domain is a simple, deterministic gridworld where the agent can move from its position to an adjacent cell. In the gridworld there are fires, pits, and a treasure. The agent gets -1 reward at each step, -500 reward for moving into a cell that contains a fire, -250 reward for moving into a cell that is adjacent to a fire, -2500 reward for moving into a pit, and +200 reward for moving into a cell that contains the treasure. The pits and the treasure are terminal states.

Our first goal was to reproduce the results from Svetlik et al. [6], which shows that transfer learning reduces the time required to learn an optimal policy. We found that transfer learning did not speed up learning when the gridworld was simple, likely because the environment was easy enough to learn with tabular Q learning. Therefore, we adjusted the position of obstacles in the environment to form a "maze" that the agent must navigate through to reach the treasure (Figure 2, left). The learning curve is shown on the right side of Figure reffig:2. Curriculum learning reduced the amount of time required to learn an optimal policy.

4.1.1 Investigating Advantage of Different Source Grid Dimensions

After demonstrating a proof-of-concept for transfer learning via reward shaping, we began investigating the effect of the size of the source grid. Specifically, we wanted to measure the "gap" in time between the transfer learning result compared to directly learning on the target task. Several metrics have been used to measure this gap, including the time to reach a given threshold δ, asymptotic performance (which compares the time to reach convergence within some threshold), and finally the jumpstart method which measures the initial performance increase on the target task from transfer [3]. We opted for the time to a given threshold δ, as this was simple both to implement and interpret. We set δ to the midpoint between the minimum and maximum reward values (averaged across a ten timestep window). Empirically, changing the threshold did not significantly alter our results, which

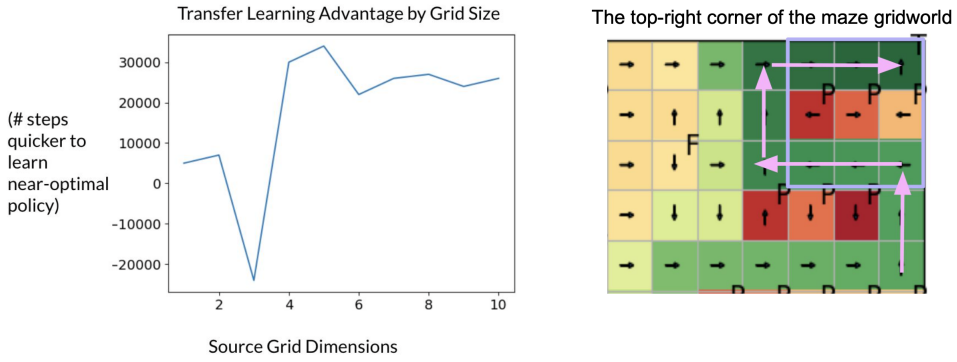


Figure 3: Left: The number of training steps more efficient to achieve a policy with reward δ with transfer learning than with direct Q-learning for various source grid sizes. Right: A close-up of the top-right corner of the maze gridworld. Cells are colored according to the learned Q-value of entering that cell, and the optimal path is shown with arrows. Note that the optimal path exits the top-right 3x3 sub-grid.

is intuitive because the gap between transfer learning and direct learning looks relatively constant throughout the learning process. For the target grid of size 10, we tried source grids from size 1 to 10. We hypothesized that source grids of medium size (4-6) would yield the greatest results, with very small grids not allowing the learner to learn enough about the environment, and large grids being too close to direct learning.

Our results in figure 3 show that all source grids of size 4 to size 10 performed quite similarly, with a source grid of size 3 performing the worst. It is an interesting result that using the entire grid as a source task (size 10) outperforms learning directly on the target. This observation may suggest a limitation of our results. Our best guess for why transfer learning outperforms Q-learning is that we randomize the initial position of the agent when learning on the source task. Even with the entire grid, randomizing initialization may create more dense reward by started the agent closer to the target. In future work we would like to control for this effect.

Additionally, by looking at the actual grid, we can see why a source grid of size 3 is particularly bad for transfer learning, which led us to discovering a limitation of grid world. In figure 3 on the right, we have outlined why a source grid consisting of the upper 3x3 grid in the maze does not lend itself well to transfer learning. We can see from the heat map that the optimal policy (overlaid with pink arrows) must exit and re-enter the 3x3 grid, in order to circumvent the three pits. However, the source task does not know that there is a way to move around these pits, since it is outside of the boundaries of the source environment. As such, the source task learns a policy that is not well transferable to the target task. To generalize, if the target task's optimal policy leaves and re-enters a given source grid, that source grid's Q values do not accurately reflect the Q values in the target task, and in fact may hinder the results. This issue is very common in the grid world domain. We will also investigate a pursuit-evasion game that does not suffer from this problem due to a state-action pair mapping from source to target tasks that is more robust.

4.2 Experiment: Q-value combination

So far, we have focused on simple curricula where the source tasks are subsets of the grid and the mapping function is just mapping the agent's position in the target task to the same position in the source task. To tackle more complex problems, we generalize Svetlik's approach by having the curriculum designer define a mapping function $f_i(s, a)$ for each source task i that maps a state-action pair in the target MDP to a state-action pair in the source MDP. The reward shaping potential function, ϕ_i, which represents information learned from source task i, now becomes

$$\phi_i(s_t, a_t) = Q_{s_i}(f_i(s_t, a_t))$$

Designing a useful mapping function requires domain-specific knowledge, but that is not a major limitation because often the curriculum is hand-designed anyway. Once we have the mapping function, if there are multiple source tasks we also define an aggregation function $g(\phi_1(s_t, a_t), ..., \phi_n(s_t, a_t))$ that aggregates the reward shaping potential from each source task into one value to use for the target task.

We run our experiments in a pursuit gridworld domain (Figure 4a). The agent controls the predators, which move concurrently and deterministically, to catch the prey, which moves in a random direction at each step. The agent receives a reward of 0 if *both* predators are in the same tile as the prey and a reward of -1 at all other steps.

The source task has a single predator and a single prey. The target task has two predators and a single prey, so the action space is a two-dimensional vector where the first dimension specifies the first predator's move and the second dimension specifies the second predator's move.

We can use the pursuit domain to test our mapping function proposal and compare different aggregation functions when there are multiple source tasks. Even though there is technically only one source task (one predator-one prey), it is as if there are two source tasks because we define two mapping functions that each map the same state-action pair in the target task to a different state-action pairs in the source task:

$$f_{s_1}(s_t, a_t) = f_{s_1}((p_{\text{prey}}, p_{\text{pred1}}, p_{\text{pred2}}), (a_{\text{pred1}}, a_{\text{pred2}})) = ((p_{\text{prey}}, p_{\text{pred1}}), a_{\text{pred1}})$$
$$f_{s_2}(s_t, a_t) = f_{s_2}((p_{\text{prey}}, p_{\text{pred1}}, p_{\text{pred2}}), (a_{\text{pred1}}, a_{\text{pred2}})) = ((p_{\text{prey}}, p_{\text{pred2}}), a_{\text{pred2}})$$

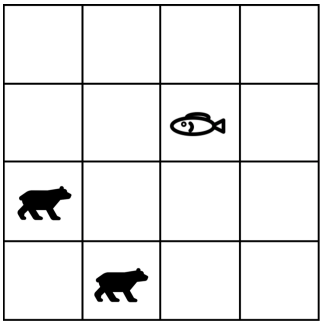
where p_{prey}, p_{pred1}, and p_{pred2} are the positions of the predators and prey. f_{s_1} essentially maps the target state-action pair to the same state-action pair in the source task excluding the second predator, and f_{s_2} does the same but excluding the first predator. To then combine information from these mapping functions into a value we can use to initialize learning on the target MDP, we experimented with the following aggregation functions:

$$g_1(\phi_1, \phi_2) = \text{max}(\phi_1, \phi_2)$$
$$g_2(\phi_1, \phi_2) = \text{mean}(\phi_1, \phi_2)$$
$$g_3(\phi_1, \phi_2) = \text{min}(\phi_1, \phi_2)$$

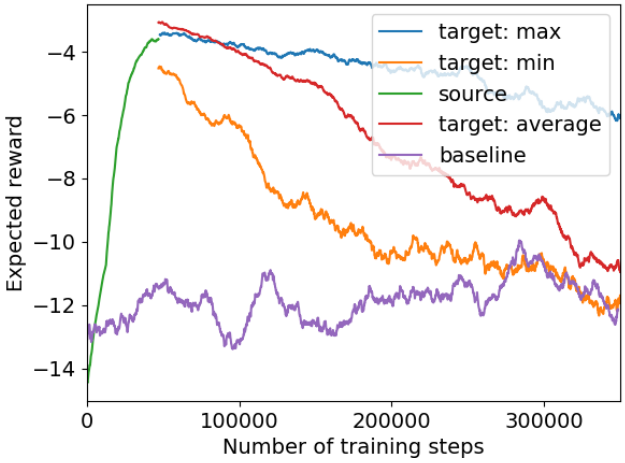
Figure 4b shows our results. The baseline is tabular Q learning with two predators. We see that this task is much harder than the gridworld - even after 350,000 steps of training the baseline doesn't learn. The source task with only one predator learns an optimal policy very quickly. Unlike in the gridworld experiment, the transferred reward shaping immediately produces a high reward policy in the target task for all three aggregation functions.

However, after more training time the performance begins to tank for all aggregation functions. Eventually, the optimal policy should be found and the learning curve will go back up because Q-learning is guaranteed to converge. We're not completely sure why performance drops. Our working theory is that even though the transferred knowledge produces a nearly optimal policy, the magnitude of the Q values is too large. This is because the source task with only one predator is solved more quickly, so the Q value at each state is higher. During Q-learning on the target two predator task, the Q values are lowered, which temporarily disrupts the policy. In the future, we could explore using a scaling factor to adjust the Q values in the target task. Reward shaping is invariant to scalar transformation to the reward function, so this wouldn't affect the optimal policy.

Our hypothesis was the that the `mean` function would perform the best because we need both predators to reach the prey, and `mean` would prioritize movements of both predators. In our experiments, the `max` function performed the best and `min` performed the worst. It makes sense that `min` would



(a) Pursuit is a 4x4 gridworld where prey (depicted as fish) move randomly and predators (bears) are controlled by the agent. Both predators must simultaneously find the prey.



(b) Learning Curve for Pursuit, averaged across five trials

Figure 4: The pursuit domain

lead to bad performance because it doesn't encourage the closer predator to approach the prey. It's not obvious to us why `max` performed so well because `max` doesn't encourage the further away predator to approach the prey and both predators need to reach the prey for a successful catch.

5 Conclusion

We demonstrated that using curriculum learning and reward shaping is an effective method of speeding up learning on complex tasks. We first demonstrated that reward shaping and curriculum learning can help combat the problem of sparse reward in a maze-like gridworld task. However, we noticed that the curricula had to be designed carefully and could in some cases even slow down learning. We then attempted to generalize the approach by defining the concept of a function that maps target task state-action pairs to source task state-action pairs in order to specify how to transfer the learned values. Using the mapping function we designed, we showed that we can aggregate information from two source tasks to speed up learning on a predator-prey gridworld task. In doing this we experimented with different methods of aggregating the potential from the source tasks, using min, max, and mean in different experiments.

Reward shaping and curriculum learning is still a fairly brittle method for speeding up learning, as source tasks must be carefully chosen and closely match the target task. In future work, we would hope to develop a more robust approach to transferring knowledge that doesn't rely on as strict a mapping from source to target task.

References

[1] Mykel J. Kochenderfer, Tim A. Wheeler, and Kyle H. Wray. *Algorithms for Decision Making*. MIT Press, 2022.

[2] Adam Daniel Laud. *Theory and application of reward shaping in reinforcement learning*. University of Illinois at Urbana-Champaign, 2004.

[3] Sanmit Narvekar, Bei Peng, Matteo Leonetti, Jivko Sinapov, Matthew E. Taylor, and Peter

Stone. Curriculum learning for reinforcement learning domains: A framework and survey, 2020.

[4] Andrew Y Ng, Daishi Harada, and Stuart Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *Icml*, volume 99, pages 278–287, 1999.

[5] Stuart Russell and Peter Norvig. Artificial intelligence: a modern approach. 2002.

[6] Maxwell Svetlik, Matteo Leonetti, Jivko Sinapov, Rishi Shah, Nick Walker, and Peter Stone. Automatic curriculum graph generation for reinforcement learning agents. 2017.