Reasoning and Learning in Real-World Systems for Long-Term Autonomy (LTA)
Papers from the AAAI 2018 Fall Symposium

Partial Policy Re-use in Connected Health Systems

Matthew Saponaro and Keith Decker

Abstract

We examine Probabilistic Partial Policy Reuse (PPR) for the purposes of developing tailored coaching strategies in the Coach-Trainee Problem (CTP). Policy reuse (PR) aims to improve a reinforcement learning agent by guiding exploration with past similar problems' learned policies. PPR extends probabilistic policy reuse that transfers only relevant parts of a policy for new problems. We explore PPR in the context of a human CTP where a coaching agent must develop a coaching strategy for the human trainee in order for the trainee to efficiently solve their problem (e.g. lose weight). In human CTPs, coach training data is limited because collecting too much data may annoy/discourage/harm the human trainee. In this paper, we present a decision tree-based algorithm, DT-partition, to identify partitions in the state space based on the problem's features, and also examine the effects of grouping problem meta-data (i.e. pruning the decision tree) on CTP performance. Particularly, we demonstrate that PPR improves task library generation and expert policy utilization compared with policy reuse.

Introduction

Wearable devices, such as smart watches, allow people to monitor their daily behaviors. Current population estimates of accelerometer-derived data indicate nearly 90% of US adults are insufficiently active and are sedentary for over half of all waking hours (55% or 7.7 hours/day) (Tucker, Welk, and Beyler 2011; Troiano et al. 2008; Matthews et al. 2008). Smart coaches, hosted/integrated with wearable devices, can improve physical activity by nudging their users to do exercise. Unfortunately, most commercial smart coaches only provide general intervention strategies and don't make use of the available information specific to their user. For example, Fitbit provides nudges 15 minutes prior to the end of each waking hour if the wearer does not walk more than 250 steps. Instead, an effective smart coach should take the context of the wearer into consideration and provide nudges when the wearer is receptive to the nudge. To create this tailored intervention strategy (i.e. determining when to nudge), a smart coach may learn when their user is receptive. We refer to learning the intervention strategy as the coach-trainee problem (CTP). For the purposes of this paper, we assume learning is done via reinforcement on the human responses. In order to learn the intervention strategy, the smart coach can probe the human user with a nudge and observe the behavioral response. Learning human behavior in this scenario is challenging because too many unsuccessful nudges may result in the smart coach being ignored or not worn. However, because there are similarities across humans, we can consider solving a single CTP by using information from different, previously solved CTPs (i.e. multi-task learning).

In this paper, we examine probabilistic policy reuse (PR) where past policies are used to guide the exploration of a new, never-before-seen problem. Unfortunately, as seen with many multi-task learning problem solutions, reusing past policies may yield negative transfer. In the human CTP, negative transfer may result with significant consequences (e.g. non-wear). PR attempts to avoid negative transfer by developing a core library of policies, then uses a policy from the library with a probability based on the utility it achieves annealed over time. In many multi-task learning scenarios, aspects of learning problems may be associated with a set of common features. In human coaching domains, humans with common traits often share similar behaviors. For example, many graduate students stay up late, while most preschoolers go to bed early. In a reinforcement learning context, these problem features may affect the underlying reward function or transition model of the problem. For the purposes of this paper, we assume that our CTP problem features are the trainee's accessibility to the gym, park, and unhealthy food options which are known contributors to a person's exercise habits. The set of problem features in a particular multi-task learning scenario are specific to the domain. For example, in a Pacman learning environment, the problem features may be the number and difficulty of the ghosts. In human coaching domains, parts of an intervention strategy may be useful to many different types of behavioral profiles but the full strategy may include some useless or harmful nudges to particular individuals. Through an indirect relationship, the solutions to learning individuals' coaching strategies are influenced by these problem features.

In this paper, we examine the effects of problem features on our CTP reinforcement learning problem solutions in order to more effectively develop smart coaches. First, we develop an algorithm to identify the subset of states controlled by the problem features (i.e. state space partitions) based on previously solved problems. Second, we develop a library of partial policies, where a partial policy is a policy for a sub-

set of states. Lastly, we examine how to integrate the partial policy library into the exploration of a new problem. The key benefit of partial policy reuse is that it captures the most relevant parts of the policy, rather than using irrelevant parts of a whole policy that resulted in net positive transfer. Additionally, completed partial policies could be used immediately without learning the whole policy. Finally, the decision trees generated by our algorithm could be initially developed by a human expert rather than computed strictly from the source tasks as presented in this paper. A human expert (e.g. fitness coach) with domain specific knowledge (e.g. what types of individuals respond well to different intervention strategies) can effectively provide initial solutions. In the human coaching domain, initial human data may be scarce so a human expert can effectively jump start the proposed algorithms (i.e. build the decision trees for DT-partition).

The remainder of this paper is outlined as follows: In section 2, we discuss the related work done in multi-task learning. In section 3, we discuss our problem definition. In section 4, we introduce the specific problem that we are trying to address. In section 5, we discuss our partial policy reuse algorithms. In section 6, we describe our experimental setup, results, and analysis. Finally, we conclude our work and discuss our next steps in section 7.

Related Work

In many multi-task learning domains, meta-data regarding problem features may describe the types of problems previously learned (Sinapov et al. 2015; Rosman, Hawasly, and Ramamoorthy 2016). For example, in the coaching domain, occupation and BMI may influence a person's ability to do moderate exercise throughout the day (e.g. waitresses walk around the restaurant while at work whereas graduate students must sit in front of a computer all day). Sinapov et al. use multi-task meta-data to develop a M5 model tree to estimate the transfer benefit from source to target task (Sinapov et al. 2015). Policies of the highest estimated benefit source tasks were directly used for the targets. Unfortunately, their approach requires a significant amount of data (i.e. solved problems) to develop their trees which makes it infeasible in the human coaching domain. In this paper, we attempt to use the problems' meta-data to a develop decision tree for each state to identify which parts of the solutions' (i.e. partial policies) state space are controlled by which problem features and thus useful for transfer. In order to address the training data issue, we extend Fernandez et al.'s work (Ferndez, Garca, and Veloso 2010). Fernandez et al. develop a policy reuse algorithm for guiding policy exploration for reinforcement learning problems; however, Fernandez et al.'s problems do not have meta-data regarding problem features. Particularly, policy reuse uses a Boltzmann exploration strategy on past solved policies weighting the use of a policy based on how well it did to solve the new problem. In this paper, we address a common issue found in both of these works—irrelevant/not useful parts of the source tasks' policies are being transferred. Because our state space partitioning algorithm identifies relevant parts of the solution based on the problem features, we improve transfer effectiveness. Furthermore, we extend Fernandez et al.'s Boltzman exploration to partial policies which then improves training data utilization. Finally, we extend Fernandez et al.'s algorithm to identify the core policy library to identify a core partial policy library.

In reinforcement learning scenarios, deep architectures typically require substantial data to train (Glatt, Da Silva, and Costa 2016). In human coaching domains, training data is limited. Furthermore, existing deep architectures require completed solutions prior to transfer. Deep architectures like ADAAPT and Actor-mimic network use neural networks to determine how to do transfer; however, decision trees provide meaningful descriptions that a domain expert (e.g. human fitness coach) may utilize to jump start agent learners (Parisotto, Ba, and Salakhutdinov 2015; Rajendran et al. 2015). Additionally, attribution of problem features to solutions can be easily identified and later used for feedback to the human fitness coaches to update then guide coaching agent learners' domain knowledge (both jump start and online).

In this paper, we make several assumptions regarding the human model that are commonly found in the literature (Saponaro, Wei, and Decker 2017; Muntaner, Vidal-Conti, and Palou 2015; Pavel et al. 2015). The goal for this coaching system is to provide a tailored intervention strategy for each trainee since general strategies tend to do poorly (op den Akker, Jones, and Hermens 2014). Trainees in this system are assumed to be either in the contemplation, preparation, or action phases of the trans-theoretical model of behavior change where trainees are thinking about and/or attempting to make a behavioral change by educating or taking action in their exercise regimen (Prochaska 2013). Furthermore, we provide trainees with just-in-time and adaptive nudges precisely when trainees are susceptible for change (in the moment), rather than scheduled. We consider trainees persisting in free-living conditions, as opposed to laboratory controlled environments (Suay and Chernova 2011). Furthermore, in this paper, we only consider when to provide nudges for activity. We do not consider the intention, content, or representation of the nudge due to the complexity of the system (op den Akker et al. 2015). Instead of modeling the fine-grained detail such as GPS location and heart rate, we abstract the human state into general categories that are meaningful across trainees, but still relevant and useful. For example, numerical GPS location would be categorized meaningful contexts such as park and gym. In this paper, we consider the impulse model proposed by Hoffman et al. (Hofmann, Friese, and Strack 2009). In Hoffman et al., humans were effectively modeled by blending a habit MDP and impulse MDP together. Particularly, trainees will either react or not to nudges to do exercise and learn the correct evaluations of states due to being nudged. Furthermore, we assume our trainees follow the markov assumption; though, a semi-markovian model can be studied to improve accuracy (Pavel et al. 2015), we chose to simplify our human models to an MDP in order to compare our work with the other machine learning literature.

Problem Definition

We demonstrate our contributions in a multi-task reinforcement learning scenario as described by Sinapov et al. (Sinapov et al. 2015). We assume the underlying problem is represented as a Markov Decision Process (MDP). An MDP is a tuple, $\langle S, A, T, R \rangle$, where S is the set of states $S = \{s_1, s_2, s_3, \ldots, s_n\}$, A is the set of actions, T is the stochastic transition function $T : S \times A \times S \rightarrow \mathbb{R}$, and R is the reward function $R : S \rightarrow \mathbb{R}$. The agent learner does not have access to T and R. We assume our reinforcement learning problems have the same domain, $D = \langle S, A, T \rangle$ and differ only by reward function. A particular reinforcement learning problem is defined as $\Omega_i = \langle D, R_i \rangle$. Each Ω_i is associated with a feature vector, $F_i = [f_1, f_2, \ldots f_m]$, such that there is an unknown and stochastic relationship between F_i and R_i.

An episode, k, of Ω_i starts with the agent located in an initial state and ends when the agent reaches a goal state (terminal) or after completing H steps. In this paper, the agents' goal is to maximize the expected average reinforcement per episode, W. This goal definition is taken from the defining policy reuse paper (Fernndez, Garca, and Veloso 2010):

$$W = \frac{1}{K} \sum_{k=0}^{K} \sum_{h=0}^{H} \gamma^h r_{k,h} \tag{1}$$

where γ is the discount factor and $r_{k,h}$ is the reward signal received from step h in episode k. In our specific application area, it is important to learn and receive positive reward early since the human may become annoyed quickly. In order to achieve its goal to optimize W, the agent learner must develop a policy $\pi : S \rightarrow A$ the dictates what action to be taken in each state. In this paper, we assume there are a set of source tasks, $T_{\text{source}} \subset \Omega$, that have been completely solved (i.e optimal policy) and a set of target tasks that have not been solved $T_{\text{target}} = \Omega - T_{\text{source}}$. In this paper, we examine developing the policy through Q-learning, though any learning algorithm to create a policy could be implemented.

The goal of Policy Reuse is to use solutions of previously solved tasks, to bias the exploration when learning the action policy of a new task in the same domain (Fernndez, Garca, and Veloso 2010). In this paper, we extend Fernandez and Velosa's policy reuse library and exploration bias strategy for partial policies. Particularly, we aim to build a partial policy library that builds a policy library for different partitions of the state space. We formally define the partial policy library as $L = \{L_{S_{p_1}}, L_{S_{p_2}}, L_{S_{p_3}}, \ldots, L_{S_{p_l}}\}$ where $L_{S_{p_i}}$ contains a library of core partial policies associated with the states $S_{p_i} \subset S$. We note that $S_{p_i} \in S_p$ is a partition of the state space and that all partitions have unique states (i.e. $S_{p_i} \cap S_{p_j} = \emptyset, \forall i \neq j$). The contributions of this paper explore how to effectively create and use this partial policy library (i.e. PPR). Particularly, we develop a library of relevant partial policies, $L_{S_{p_i}} = \{L_1^{S_{p_i}}, L_2^{S_{p_i}}, \ldots, L_\ell^{S_{p_i}}\}$ where $L_\ell^{S_{p_i}}$ is the set of core partial policies relevant to agent group ℓ in partition S_{p_i}. Later in the paper, we describe how to partition the state space (i.e. determine S_p) based on F for the purposes of identifying related solution parts (i.e. DT-partition). We then describe how to build a core partial policy library, L.

Algorithm 1 DT-Partition

procedure DT-PARTITION($T_{\text{source}}, S,$)
  for $s_i \in S$ **do**
    $dt_i \leftarrow$ buildTree (s_i, T_{source})
  end for
  $D \leftarrow \emptyset$ ▷ initialize state partition trees
  $S_{\text{r}} \leftarrow S$ ▷ unpartitioned states
  $S_p \leftarrow \emptyset$ ▷ initialize state space partitions
  d $= 0$ ▷ initialize partition index
  while $S_{\text{r}} \neq \emptyset$ **do**
    $s_k \leftarrow S_{\text{r}}$.pop ()
    $S_{p_d} \leftarrow S_{p_d} \cup \{s_k\}$ ▷ initialize partition S_{p_d}
    $D \leftarrow D \cup \{dt_k\}$ ▷ initialize partition's DT
    for $s_i \in S_{\text{r}}$ **do**
      if DT-equal(dt_i, dt_k) **then**
        $S_{p_d} \leftarrow S_{p_d} \cup \{s_i\}$ ▷ add s_i to partition
        $S_{\text{r}} = S_{\text{r}} - \{s_i\}$
      end if
    end for
    $S_p \leftarrow S_p \cup \{S_{p_d}\}$
    $d = d + 1$ ▷ increment partition index
  end while
  return S_p, D
end procedure

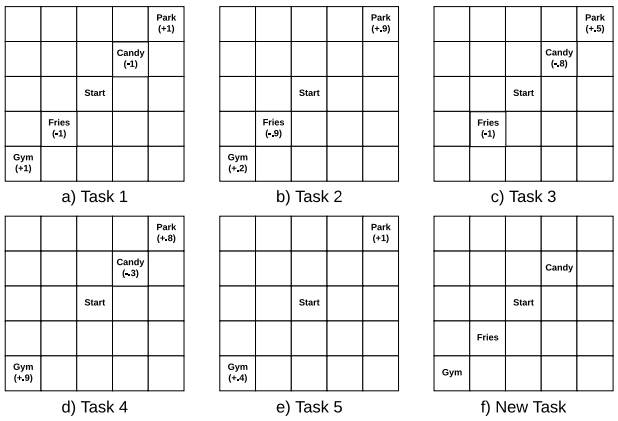

Figure 1: Grid-based CTP where a) through e) are source tasks and f) is the target task.

Grid-CTP

We demonstrate the effectiveness of partial policy reuse in a grid-based coach-trainee problem (CTP) domain where a coach must learn where an individual is most likely to do exercise while not being distracted by their surroundings.

In our grid-based coach-trainee problem, the underlying MDP can be modeled such that the states, S, are the grid locations that the trainee can be located, the actions are

the nudged direction that the trainee should go, the transition model represents the trainee's likeliness to follow the coach's nudge, and the reward function, R, is the health benefits achieved by going to a particular location. All trainees are assumed to have the same transition model—follow the coach's advice 90% of the time and wander randomly the other 10%. The reward function's are different per problem and related to the problem features. In this paper, we examine features related to the trainee's surrounding location. Particularly, we examine the following trainee's accessibility features: park, gym, candy store, and a fry shop. Candy shops and fry shops are distractions to the health benefits achieved at parks and gyms. In figure 1, we show 6 grid-CTPs. Each grid-CTP have different problem features. For example, in figure 1 a), the trainee has access to the gym, park, fry shop, and candy store, but in figure 1 b) the trainee doesn't have access to a candy store. Due to the variance in individual trainees, we assume rewards take a uniform distribution where health benefits are between 0 and 1 and distractions are between -1 and 0. If a trainee has no access to a community utility (e.g. gym), then no reward is given for that state. For an example, in figure 2 d), the trainee has no access to a gym. An episode represents a day of the trainee and the trainee may only go to one community utility; thus, rewarding states are terminal. We assume there are only 18 steps per episode representing that a person only has 18 waking hours per day. Furthermore, we assume we have a collection of previously solved CTPs where the coach policy is complete. In this paper, we attempt to use this past collection of coach policies to quickly learn a new trainee's coach's policy.

We make several assumptions regarding CTP that do not affect the contributions of this paper. Particularly, we assume the transition model is fixed meaning that the trainee is not changing any behaviors due to incomplete or incorrect perceptions of the world. For example, in the full-CTP, a trainee may initially highly value eating french fries because they taste good, but when the coach indicates french fries are unhealthy, the trainee would update its views on french fries and avoid them. Furthermore, the coach does not model the trainee's readiness to receive an intervention. For example, a trainee may be busy in the morning and unwilling to do exercise, and therefore ignore whatever the coach says. Finally, we limit the problem features in this paper to the accessibility of different physical locations. In the full CTP, problem features include occupation, body mass index (BMI), age, social status, community utilities, etc. Though we do not examine the full problem, the algorithms in this paper can generalize to the full problem.

Partial Policy Reuse

In policy reuse, a source task policy is used to guide the exploration when solving a target task. We argue that the source task's full policy may include irrelevant/harmful parts in the target's problem. For example, all athletes may be susceptible to nudges in the morning; however, some find it beneficial to do light exercise at night, while other athletes find it exhaustively harmful. In this example, there may be a partition for the day coaching strategies and a partition for the night coaching strategies. Additionally, we argue that all source tasks may not be relevant to the target task's solution. For example, runners like to run at night because the air is cool, but weight lifters do not like to lift weights at night because the gym is dirty from being used during the day. In this example, using a coaching strategy for a jogger on a runner may make sense since both like to do exercise at night, but using the jogger coaching strategy for a weight lifter may result in negative transfer. Partial policy reuse aims to mitigate these issues by determining what are the relevant parts of the solution to transfer (i.e. DT-partition) and what are the relevant source task solutions to use when transferring these partial solutions (i.e. partial-policy library).

DT-Partition

This section explains the DT-partition algorithm. DT-partition aims to partition the state space based on the similarity of solutions for problems with the same features in order to identify parts of solutions that are relevant for transfer. We define a state space partition as a set of states $S_{p_i} \subset S$ such that a single partition has no overlap of states with any other partition. The following DT-partition algorithm partitions the states based on the agreement of the source task policies. See algorithm 1.

Algorithm 2 DT-Compare

```
1:  procedure DT-EQUAL(dt_i, dt_j)
2:      if nodeType(dt_i) ≠ nodeType(dt_j) then
3:          return False                          ▷ leaf vs. internal node
4:      else if isLeaf(dt_i) then
5:          return dt_i.tasks == dt_j.tasks
6:      else
7:          if dt_i.splitAttribute ≠ dt_j.splitAttribute then
8:              return False
9:          else
10:             isEqual ← True
11:             for chld ∈ children do
12:                 subEql ← DT-equal(dt_i.chld, dt_j.chld)
13:                 isEqual ← isEqual && subEql
14:             end for
15:             return isEqual
16:         end if
17:     end if
18: end procedure
```

For each state, s_i, develop a decision tree, dt_i, using $\Omega_t \in T_{\text{source}}$ as the instances such that F_t are the features and $\pi_{\Omega_t}(s_i)$ is the class value. For the purposes of this paper, we use the C4.5 decision tree algorithm. Then, group states together (i.e. label as a partition) such that the decision trees are equivalent in structure. We assert that two decision trees are equivalent if the internal nodes have the same split points and the source tasks that reach the leaves are the same. See algorithm 2. We note that the leaf classifications may be different across trees, but as long as the instances that reach the node (i.e. Ω) and their class values are in agreement (i.e. $\pi_{\Omega_t}(s_i) = \pi_{\Omega_u}(s_i)$), we consider the trees equivalent.

In our experiments section, we analyze how pruning the tree structure (i.e. decreasing number of partitions) effects transfer learning outcome. We note that if a tree were completely pruned, the algorithm would produce a single partition equivalent to S and thus replicate policy reuse.

Partial-Policy Library

The partial policy library consists of relevant core policies for each partition of the state space for the purposes of effectively transferring knowledge. We describe how to develop a partial policy library in algorithm 3.

Algorithm 3 Build-Partial-Policy-Library

procedure BUILD-PP-LIBRARY($T_{\text{source}}, S, \theta$)
  $L \leftarrow \emptyset$
  $S_p, D \leftarrow$ DT-partition (T_{source}, S)
  for $i \in \{0, \ldots, |S_p|\}$ **do**
    $L_{S_{p_i}} \leftarrow \emptyset$
    for $\ell \in D_i$.leaves **do**
      $L_l^{S_{p_i}} \leftarrow$ add-core-PP $(S_{p_i}, \ell.\Omega, \theta)$
      $L_{S_{p_i}} \leftarrow L_{S_{p_i}} \cup \{L_l^{S_{p_i}}\}$
    end for
    $L \leftarrow L \cup \{L_{S_{p_i}}\}$
  end for
end procedure

In our algorithm, we partition the state space using a partitioning algorithm like the one previously discussed. For each partition S_{p_i}, we use the corresponding partition's decision tree structure, D_i, to determine the source tasks for each relevant group ℓ. Instances at each leaf in D_i are considered within the same relevant source task group. In our experiments described later, we examine pruning the tree's structure to show the effects of available source tasks and relevancy for transfer.

As described in algorithm 5, using the relevant source tasks at each leaf, $\Omega_\ell^{S_{p_i}}$, we build the core partial library for $L_\ell^{S_{p_i}}$. For a particular source task, Ω_m, we add its corresponding policy, π_m, to the library if the percent increase in expected performance (i.e. $\hat{w}_m$) of using its policy compared with the expected performance of the best policy in the library is greater than a threshold, θ. In order to estimate the W, we use the task's Q-values. It is non-trivial to directly compute the partial policies expected performance because the performance is dependent on the full policy.

We note that when we fully prune the tree's structure when determining source task selection, all source tasks are used. Furthermore, when we prune both the state space partition tree and the source task selection tree, we reduce our partial policy reuse algorithms to the defining policy reuse algorithm as defined in (Fernndez, Garca, and Veloso 2010). Furthermore, though we present a decision tree based algorithm that uses agreement between policies, other optimizations that better partition the state space and identify relevant source tasks may be utilized.

Algorithm 4 Ω_{new} Learning from the partial policy library

procedure LEARN($\Omega_{\text{new}}, L, K, H, v, \Delta\tau$)
  $Q_{\text{new}}(s, a) = 0, \forall s \in S, a \in A$
  $U_{S_{p_i},n} = 0, \forall i : S_{p_i} \in S_p, \forall n : \pi_n^{S_{p_i}} \in L_\ell^{S_{p_i}}$
  $W_{s_{p_i},n} = 0$
  $\tau = 0$
  for k = 1 to K **do**
    $s \leftarrow s_{\text{start}}$
    $\phi \leftarrow \phi_{\text{init}}$
    $\pi_{S_{p_i}} \leftarrow \pi_{S_{p_i},n}$ with probability $\frac{e^{\tau \cdot W_{s_{p_i},n}}}{\sum_o e^{\tau \cdot W_{s_{p_i},o}}}$
    $W_k = 0$
    for h = 1 to H **do**
      $S_{p_i} \leftarrow S_{p_k}$ s.t. $s \in S_{p_k}$
      With probability $\phi, a = \pi_{S_{p_i}}$
      With probability $1-\phi, a = \epsilon$-greedy (Ω_{new})
      $s' \leftarrow$ next state, $r_{k,h} \leftarrow$ reward signal
      $W_k \leftarrow W_k + \gamma^h \cdot r_{k,h}$
      update Q_{new} and π_{new}
      $\phi \leftarrow \phi \cdot v$
      $s \leftarrow s'$
    end for
    $\tau \leftarrow \tau + \Delta\tau$
    $U_{S_{p_i},n} \leftarrow U_{S_{p_i},n} + 1, \forall S_{p_i} \in S_p$
    $W_{s_{p_i},n} \leftarrow \frac{W_{s_{p_i},n} \cdot (U_{s_{p_i},n} - 1) + W_k}{U_{s_{p_i},n}}, \forall S_{p_i} \in S_p$
  end for
end procedure

Algorithm 5 Add Core Partial Policies

procedure ADD-CORE-PP($S_{p_i}, \Omega_l^{S_{p_i}}, \theta$)
  $L_l^{S_{p_i}} \leftarrow \emptyset$
  for $\Omega_m \in \Omega_l^{S_{p_i}}$ **do**
    $\pi_m \leftarrow \Omega_m.\pi$
    $\hat{W}_{\pi_m} \leftarrow \sum_{s_j \in S_{p_i}} Q_m(s_j, \pi_m(s_j))$
    $\hat{W}_{\text{best}} = \max_{\pi_k \in L_l^{S_{p_i}}} \sum_{s_j \in S_{p_i}} Q_m(s_j, \pi_k(s_j))$
    if $\hat{W}_{\pi_m} - \hat{W}_{\text{best}} > \theta \left|\hat{W}_{\text{best}}\right|$ **then**
      $L_l^{S_{p_i}} \leftarrow L_l^{S_{p_i}} \cup \{\pi_m\}$
    end if
  end for
  return $L_l^{S_{p_i}}$
end procedure

Partial-Policy Reinforcement Learning

In this section, we describe how an agent may use the partial policy library to learn a policy for a new task Ω_{new}. See algorithm 4. Partial Policy reinforcement learning extends Policy Reuse reinforcement learning by simultaneously using multiple partial policies,$\pi_{S_{p_i}}$, to guide the agent learner for a particular episode, k.

For a particular episode, k, an agent chooses a partial policy to execute in each partition, S_{p_i}, of the state space. A partial policy, $\pi_{S_{p_i},n} \in L_\ell^{S_{p_i}}$ is chosen from the library based on the agent learner's performance, W. A particular partial policy is likely to be used when the agent has good performance and unlikely to be used when the agent has poor performance with respect to other partial policies in $L_\ell^{S_{p_i}}$. In the proposed partial policy reinforcement learning algorithm, the partial policy's individual performance, $W_{S_{p_i},n}$, is effected by the combined performance of all partial policies; however, $W_{S_{p_i},n}$ will eventually converge to the true evaluation after collecting enough data. We believe this problem of attributing individual partial policy performance is an open area for researchers to address in order to improve partial policy usages. For a particular step, h, the agent can either use its own policy, π_{new}, or use the expert partial policy, $\pi_{S_{p_i}}$. When the agent learner uses the expert partial policy, the agent passively learns and updates its Q-table based on the expert's action and the reward signal the learner received. The learning agent anneals the temperature parameter, ϕ, that controls the probability of using the expert's policy with respect to the number of steps taken. Additionally, the agent learner anneals the reuse parameter, τ, over the episodes and increases the likelihood to choose the best partial policies based on their performances.

In domains with limited source tasks, our partial policy library algorithm may result in state partition libraries (i.e. $L_\ell^{S_{p_i}}$) that do not have any expert policies. In this particular scenario, either the tree used to generate the libraries should be pruned to impose a larger source task library or the agent learner can simply use its own policy without the guidance of an expert.

Experiments

We demonstrate the effectiveness of our algorithms in the Grid-CTP domain. Particularly, we demonstrate the effects of our state space partitioning algorithm, DT-partition, on coach learning performance and the effects of the relevance of source tasks on our coach's performance. The algorithms presented in this paper effectively target two questions: 1) How much of the state space can be transferred in order to effectively guide the new learner? 2) Which tasks are relevant enough to be used when guiding the new learner? In order to demonstrate that our algorithms address these questions, we present the following experiments. The first experiment prunes the trees used to make the state partitions in DT-partition. By pruning these trees, the partial policies become larger (i.e. $|S_{p_i}|$ increases). The second experiment prunes the decision trees after creating the state space partitions; thus, increasing the number of the expert policies in the source task library.

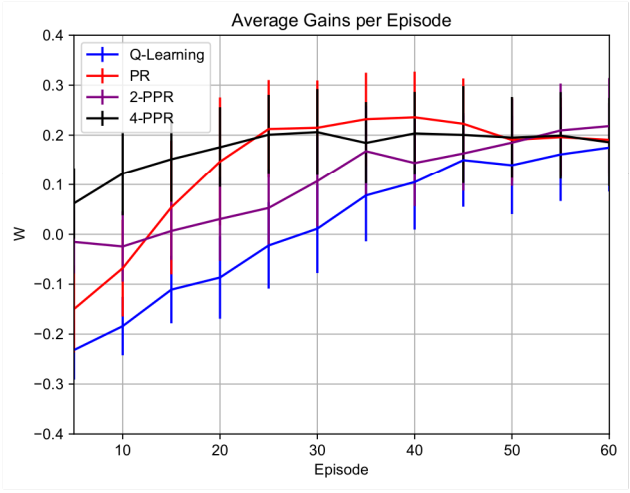

Figure 2: PPPR state space partition decision tree was pruned to have a height of 2 (i.e. 2-PPR) and 4 (i.e. 4-PPR)

We generated 160 source tasks—10 repetitions of all combinations of our 4 Grid-CTPs' problem features (i.e. accessibility to gym, park, candy store, fry shop), where the reward functions of each task take on a uniform distribution between 0 and 1 for states with health benefits (i.e. gym and park) and 0 and -1 for health distractions (i.e. candy and fries). We then use these 160 source tasks as experts to guide a coach learner to solve a new task. In order to efficiently use these source tasks, we first build the partial policy library as described above in algorithm 3. In order to add a partial policy to the library, the partial policy needs to have a 10% performance increase (i.e. $\theta = .1$). We then use the partial policy library to execute a coach learner on a new problem as described in algorithm 4. The new problem has accessibility to the gym, fry shop, and candy store, but no accessibility to the park. Our coach learner's initial ϕ is 1, and cools down by 1% each step (i.e. $v = .01$). The coach expert preference temperature parameter, τ increases linearly by 1 unit every episode (i.e. $\Delta\tau = 1$); thus $\frac{1}{\tau}$ decreases and prefers better experts in future episodes. We execute the learning phase for the agent coach 100 times and take the average performance across 100 runs for each episode.

Increasing the State Partition Size

In this experiment, we build the full decision trees for each state used to partition the state space, then prune them to heights of 0, 2, and 4 before partitioning the state space in algorithm 1. By pruning the trees, we decrease the entropy of agreement of policies in the leaves. We note that when we prune the trees to a height of 0, the algorithm reduces to policy reuse. We show the performance of the coach learning using partial policy reuse when the heights of the trees are 2 (i.e. 2-PPR), and 4 (i.e. 4-PPR) in figure 2.

We see that partial policy reuse, when building the full tree, does best compared against standard Q-learning and Policy Reuse. Interestingly, the relationship between the

height of the tree and the performance is not linear. Particularly, the decision trees with height 2 do worse than when the heights are 0 and 4. We hypothesize that the decision trees of height 2 perform poorly with respect to the other heights because the useful amount of knowledge per partition was low. Particularly, we believe there were too many decisive states (i.e. parts of trajectories that led to reward), yet not enough useful knowledge gained from the experts in these partitions.

Increasing the Quantity of Source Tasks in Library

In this experiment, we partition the state space using algorithm 1; however, after we've determined the state space partition, we prune the decision trees to heights of 0, 2, and 4 before developing the core partial policy library. For a particular partition, pruning the tree increases the number of source tasks used in the partial policy library for that partition. When plotting the results of average performance per episode, We get similar results as shown in figure 2 where the fully developed tree performs best, while the height of 2 tree performs only better than Q-learning. We hypothesize the that partitioning algorithm may not be yielding effective partial expert policies in some of the partitions.

Conclusion

In this paper, we demonstrate the effectiveness of probabilistic partial policy reuse in the CTP domain. In human learning domains where training data is limited and costly, agents need to quickly learn a solution in order to not be seen as annoying or irrelevant (i.e. low time to threshold). Partial policy reuse allows agent learners to achieve this by partitioning the state space and using relevant partial policies to solve a problem. In this paper, we present novel algorithms to partition the state space, develop a core partial policy library, and use a partial policy library while learning. Our algorithms allow for agent learners to learn parts of the state space and become partial experts (i.e. source tasks) without completely solving the underlying problem. The partial policy reuse algorithm presented in this paper trades off both the amount of transferred content and the availability of source tasks with the relevance of source tasks. In this paper, we examine decision tree based solutions because domain experts (i.e. fitness coaches) may be able to manually provide a structure based on their expert opinion. For example, in the CTP domain, a physical activity coach may notice that occupation is a strong indicator of group behavior in physical activity. Particularly, nurses walk often during the day, so nudges may be useless during the day, but for graduate students, nudges may be useful during the day. Though we develop partial policies, individual partial policy performance attribution is an open question. In realistic human learning systems such as CTP, a better attribution design may be needed. The work presented in this paper attempts to improve learning efficiency in order to create tailored interventions strategies in the human coaching domain; however, our work can also be applied to other multi-task learning domains.

References

Fernndez, F.; Garca, J.; and Veloso, M. 2010. Probabilistic policy reuse for inter-task transfer learning. *Robotics and Autonomous Systems* 58(7):866 – 871. Advances in Autonomous Robots for Service and Entertainment.

Glatt, R.; Da Silva, F. L.; and Costa, A. H. R. 2016. Towards knowledge transfer in deep reinforcement learning. In *Intelligent Systems (BRACIS), 2016 5th Brazilian Conference on*, 91–96. IEEE.

Hofmann, W.; Friese, M.; and Strack, F. 2009. Impulse and self-control from a dual-systems perspective. *Perspectives on Psychological Science* 4(2):162–176.

Matthews, C. E.; Chen, K. Y.; Freedson, P. S.; Buchowski, M. S.; Beech, B. M.; Pate, R. R.; and Troiano, R. P. 2008. Amount of time spent in sedentary behaviors in the united states, 2003–2004. *American journal of epidemiology* 167(7):875–881.

Muntaner, A.; Vidal-Conti, J.; and Palou, P. 2015. Increasing physical activity through mobile device interventions: A systematic review. *Health informatics journal* 1460458214567004.

op den Akker, H.; Cabrita, M.; op den Akker, R.; Jones, V. M.; and Hermens, H. J. 2015. Tailored motivational message generation: A model and practical framework for real-time physical activity coaching. *Journal of biomedical informatics* 55:104–115.

op den Akker, H.; Jones, V. M.; and Hermens, H. J. 2014. Tailoring real-time physical activity coaching systems: a literature survey and model. *User modeling and user-adapted interaction* 24(5):351–392.

Parisotto, E.; Ba, J. L.; and Salakhutdinov, R. 2015. Actor-mimic: Deep multitask and transfer reinforcement learning. *arXiv preprint arXiv:1511.06342*.

Pavel, M.; Jimison, H. B.; Korhonen, I.; Gordon, C. M.; and Saranummi, N. 2015. Behavioral informatics and computational modeling in support of proactive health management and care.

Prochaska, J. O. 2013. Transtheoretical model of behavior change. In *Encyclopedia of behavioral medicine*. Springer. 1997–2000.

Rajendran, J.; Prasanna, P.; Ravindran, B.; and Khapra, M. 2015. Adaapt: A deep architecture for adaptive policy transfer from multiple sources. *arXiv preprint arXiv:1510.02879*.

Rosman, B.; Hawasly, M.; and Ramamoorthy, S. 2016. Bayesian policy reuse. *Machine Learning* 104(1):99–127.

Saponaro, M.; Wei, H.; and Decker, K. 2017. Towards learning efficient intervention policies for wearable devices. In *Connected Health: Applications, Systems and Engineering Technologies (CHASE), 2017 IEEE/ACM International Conference on*, 298–299. IEEE.

Sinapov, J.; Narvekar, S.; Leonetti, M.; and Stone, P. 2015. Learning inter-task transferability in the absence of target task samples. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*, 725–733. International Foundation for Autonomous Agents and Multiagent Systems.

Suay, H. B., and Chernova, S. 2011. Effect of human guidance and state space size on interactive reinforcement learning. In *2011 Ro-Man*, 1–6. IEEE.

Troiano, R. P.; Berrigan, D.; Dodd, K. W.; Masse, L. C.; Tilert, T.; McDowell, M.; et al. 2008. Physical activity in the united states measured by accelerometer. *Medicine and science in sports and exercise* 40(1):181.

Tucker, J. M.; Welk, G. J.; and Beyler, N. K. 2011. Physical activity in us adults: compliance with the physical activity guidelines for americans. *American journal of preventive medicine* 40(4):454–461.