# Domain Adaptation in Reinforcement Learning

A Thesis<br>
Presented to<br>
The Academic Faculty

by

**Srijan Sood**

In Partial Fulfillment
of the Requirements for the Degree
B.Sc. in Computer Science with the Research Option

School of Computer Science
Georgia Institute of Technology

August 2017

# Domain Adaptation in Reinforcement Learning

Approved by:

Dr. Charles L. Isbell, Jr., Advisor
College of Computing
*Georgia Institute of Technology*

Dr. Irfan Essa
College of Computing
*Georgia Institute of Technology*

Date Approved: August 1, 2017

When you have eliminated the impossible, whatever remains, however improbable, must be the truth.

*Sherlock Holmes*

Dedicated to my wonderful parents, Indra and Vinod Sood.

# ACKNOWLEDGEMENTS

I would like to thank my family — my parents Indra and Vinod Sood, and my brother Shreyas Sood — for always believing in me, supporting me and pushing me to do my best.

At Georgia Tech, I would like to thank Dr. Charles Isbell for giving me the opportunity to work in his lab and to learn from him. His advice and mentorship are invaluable. Most of all, I would like to thank his Ph.D. student, Ashley Edwards, for mentoring me and taking me under her wing. This research was conducted in conjunction with her, and this work would not be possible without her guidance and instruction.

## TABLE OF CONTENTS

# LIST OF TABLES

# LIST OF FIGURES

# Abstract

Reinforcement learning is a powerful mechanism for training artificial and real-world agents to perform tasks. Typically, one can define a task for an agent by simply specifying rewards that reflect the agent's performance. However, each time the task changes, one must develop a new reward specification. Our work aims to remove the necessity of designing rewards in tasks consisting of visual inputs. When humans are learning to complete tasks, we often look to other sources for inspiration or instruction. Even if the representation is different from our own, we can adapt our own representation to the task representation. This motivates our own work, where we present tasks to an agent that are from an environment different than its own. We compare the cross-domain goal representation with the agents representation to form Cross-Domain Perceptual Reward (CDPR) functions and show that these enable the agent to successfully complete its task.

# CHAPTER 1
# INTRODUCTION

Reinforcement Learning (RL) is an area of machine learning inspired by psychology, and its problems involve learning what to do — how to map situations to actions so as to maximize a numerical reward signal [1]. Essentially, problems are described by positive and negative numerical rewards that indicate if an agent has successfully completed a given task. The agent tries to find the optimal policy, i.e., the sequence of actions that will lead to the greatest cumulative (often delayed) reward through a combination of exploration (of the unknown state space) and exploitation (of current knowledge).

RL agents are traditionally limited to relatively low-dimensional domains in which features can be manually defined. To address this, Deep Learning (convolutional neural networks, in particular) was used in conjunction with Reinforcement Learning, to create deep Q-networks (DQN). This allow us to directly extract features from visual data and consequently learn control policies successfully [2]. However, reinforcement learning tasks heavily depend on rewards, which normally need to be specified manually, which requires extensive domain knowledge, or at least knowledge about the configuration of the domain. Therefore, we need methods to easily specify a task or goal, without accessing the underlying mechanics of an environment. This is similar to learning how to play a game by clicking around randomly, and learning how to assemble an object from videos or pictures in an instruction manual.

Previous work introduced the idea of using perceptual rewards for representing tasks visually [3], as opposed to explicit rewards. Other work has also included ways to extract features the environment to represent a task to an agent, and then learn the optimal policies to do so. Examples include adapting robotic perception from simulated to real-world environments [4], and using demonstrations along with perceptual rewards for imitation learning [5].

Our work aims to only specify the final goal visually, rather than needing to demonstrate it through a series of images or videos. We also aim to eliminate the need to manually map the agent's representation to the cross-domain example, as well as motion templates and HOG (Histogram of Oriented Gradients) feature descriptors used in pervious work [3]. We will instead utilize deep learning to learn features in the agent's domain, and the alternative domain, and then use these to find similarities between the two. These similarities shall serve as a Cross-Domain Perceptual Reward function, which we plan to use to solve tasks using standard RL approaches.

# CHAPTER 2
# RELATED WORK

Developing goals based on hand-crafted rewards can often be tedious. Our approach aims to simplify the process by providing cross-domain goal images. Other works have provided goals as images in the agent's domain [6]. Learning from demonstration allows one to provide demonstrations of how the task should be accomplished. Inverse Reinforcement Learning can be used to learn a reward from the demonstrations [7, 8], and Imitation Learning can be used to reproduce behaviors that are consistent with demonstrated policies [9]. These approaches often assume that the agent's representation is obtained from the same distribution as the target. No work to our knowledge has utilized visually dissimilar task representations to learn rewards. However, many approaches aim to learn across different domains. In particular, it is often more efficient to train in simulation and then transfer the knowledge to the real-world [10, 11, 12].

There has been a large body of work in deep learning for finding similarities across domains, generally applied to image classification, generation, and retrieval problems [13, 14, 15, 16]. There has also been work to translated images from one domain to the other, something that could be leveraged in our approach [17, 18].

Our work explores the use of such techniques in Reinforcement Learning, allowing tasks to be specified across visually dissimilar domains in the form of the desired goal, by utilizing a single image (as opposed to a series of demonstrations). We aim to learn a reward function from these cross-domain examples, and then utilize it to solve tasks using standard RL approaches.

# CHAPTER 3

# BACKGROUND

Reinforcement learning problems are described through a Markov Decision Process $\langle S, A, P, R\rangle$ [1]. The set $S$ consists of states $s \in S$ that represent the current variables of an agent's environment. An agent takes actions $a \in A$ and receives rewards $r \in R(s)$ that depend on the current state. The transition function $P(s, a, s')$ represents the probability that the agent will land in state $s'$ after taking action $a$ in state $s$. The learning approach that we use is model-free and does not have access to $P$. A policy $\pi(s, a)$ represents the probability of taking action $a$ in state $s$. Goals in reinforcement learning problems are often described solely through the reward function. An action-value, or Q-value, $Q(s, a)$ represents the expected discounted cumulative reward an agent will receive after taking action $a$ in state $s$, then following $\pi$ thereafter. We typically are interested in computing optimal Q-values:

$$Q^*(s,a) = \max_\pi \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s, a_t = a, \pi\right]$$

where $0 \leq \gamma < 1$ is a discount factor that encodes how rewards retain their value over-time.

Q-values can be estimated through tabular methods that map every state and action to a value. However, with large or continuous state spaces, it is often necessary to compute the values with function approximation. In our case, the state inputs will be images and so we use a Deep Q-Network (DQN) to approximate the value function, as it has been empirically shown to perform well with visual inputs [2].

Tasks are typically defined solely through rewards. This work aims to make the reward representation general—only the inputs should change.

# CHAPTER 4
# TECHNICAL APPROACH

The problems we aim to solve consist of visual states and have multiple possible goals for a singular task. For example, an agent's task might be to navigate a maze, but each time the goal might vary, as its start point, end point and the maze itself might change. In other words, each goal is an instance of a broader task. Rather than using manually engineered reward values (as is typically done in RL), we aim to describe the reward by showing the agent how the completed goal *looks*. For each task, this cross-domain perceptual reward function (CDPR) will be a function of the agent's state and the goal provided to it. Our approach focuses on providing goal examples to the agent from a domain (environment) visually different than its own. The benefits of this are multi-faceted. Such an approach would be useful when the task requirements are complex to craft rewards for, and if we do not have access to the agent or environment's internal configuration. Moreover, this allows us to to describe goals even if we do not know how to solve the task in the agent's domain.

To successfully provide reward from cross-domain examples, we need to have a mechanism to compare images with similar content but different representations. Standard similarity metrics such as the euclidean distance would fail at this. Convolutional Neural Nets however, have shown the capability to learn features from raw inputs, and have been heavily employed in image classification tasks. We propose a reward network that leverages these capabilities to construct a similarity metric between cross-domain images. We now formalize our approach for building these visual task descriptions, and correspondingly CDPRs.

## 4.1 Obtaining samples

We focus on tasks consisting of multiple goal instantiations. For example, an agent's *task* could be to assemble furniture, but the *goal instantiation* would specify the configuration for the piece of furniture to be built. We aim to specify goals through images that are from different domains than the agent's, for example, a drawing or a CAD model of the furniture to be built.

Consider a task that consists of $N$ possible goal instantiations $\Gamma_1, \Gamma_2, \ldots, \Gamma_N$. Given a cross-domain goal $\widehat{G}_j$., our objective is to assign goal instantiations so that $\Gamma := \widehat{G}_j$. We aim to train a deep neural network that allows specifying goals in this manner. We train the network with a semi-supervised approach. We assume for a subset of goal instantiations, that we have a single image of the goal specification in the agent's environment, $G_i$, and a corresponding image from the cross-domain environment, $\widehat{G}_i$. After training, we only use the cross-domain goal to instantiate $\Gamma$. The idea is that we may be willing to provide some pairings to learn a correspondence between the two domains, but then the agent must learn to achieve goals specified in the cross-domain representation.

## 4.2 Reward Network

### 4.2.1 Architecture

Figure 4.1 shows the network we use to learn CDPRs. The inputs to the network consist of states from the agent's environment $G_i$, and cross-domain goals $\widehat{G}_j$. The network learns features for both domains and the CDPR uses these to output a distance between the agent's state and the cross-domain goal provided to it.

The reward network is composed of two independent convolutional networks with identical architectures that encode the intra-domain features, $\Phi_{G_i}$, and the cross-domain features, $\Phi_{\widehat{G}_j}$, respectively. The CDPR is obtained by taking the dot product of these two

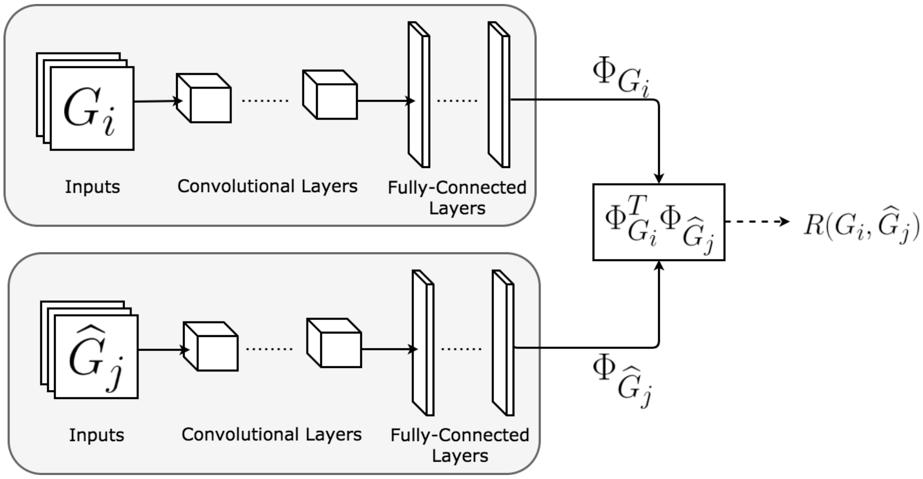


Figure 4.1: Reward Network used to learn CDPRs. Two identical networks encode features for the agent's states $G_i$ and cross-domain goals $\widehat{G}_j$, respectively. The dot product between the outputs of each encoding layer represents the CDPR, $R(G_i, \widehat{G}_j)$.

outputs:

$$R(G_i, \widehat{G}_j) = \Phi_{G_i}^T \Phi_{\widehat{G}_j} \tag{4.1}$$

This reward represents the similarity between an agent's state and a cross-domain goal. Once we obtain the CDPRs, we can use them to instantiate the reward function for the MDP, and can use standard RL approaches to solve the task. However, it is worth noting that while the CDPR remains fixed across goal specifications, it is necessary to learn a new CDPR for each type of cross-domain goal representation, as the entire set of samples $\widehat{G}_j$ would change.

### 4.2.2 Training

We train the network with a semi-supervised approach. Assume that for a given task, that we have $k$ pairs consisting of an intra-domain goal specification and a corresponding cross-domain goal: $\{(G_1, \widehat{G}_1), \ldots, (G_k, \widehat{G}_k)\}$. After training, we only use cross-domain images to instantiate goals. A good reward function would make $R(G_i, \widehat{G}_j)$ large when $i = j$ and small otherwise. For each training batch, we randomly sample two goal pairs, $(G_i, \widehat{G}_i)$ and $(G_j, \widehat{G}_j)$. The loss aims to make the reward for the matching pair, $R(G_i, \widehat{G}_i)$, be larger than the rewards for mismatched pairs, $R(G_j, \widehat{G}_i)$ and $R(G_i, \widehat{G}_j)$. If it isn't, the outputs for the loss function will be greater than $0$ and so the gradients will be penalized.

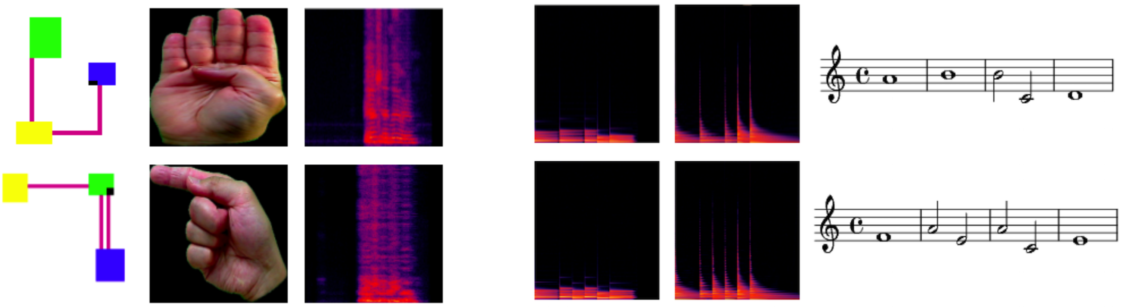
Our approach is inspired by one that utilizes deep learning to find similarities between images and their (cross-domain) spoken captions [18]. We utilize the loss function from this approach, modified appropriately for learning reward functions. Given the parameters $\theta$ of the network described in Figure 4.1, the loss can be formulated as:

$$\mathcal{L}(\theta) = \max\Big(0, R_\theta(G_j, \widehat{G}_i) - R_\theta(G_i, \widehat{G}_i) + 1\Big) + \max\Big(0, R_\theta(G_i, \widehat{G}_j) - R_\theta(G_i, \widehat{G}_i) + 1\Big) \tag{4.2}$$

We use gradient descent to optimize this loss function.

## 4.3 Tasks

We worked with two tasks, each of which had two different cross-domain goal representations apart from the agent's domain.



(a) Maze Domain

(b) Music Domain

Figure 4.2: Domains used for each RL task. a) In the Maze domain, the agent's task is to navigate to a specific room. The goal specifies the room the agent needs to navigate to. For the cross-domain goals, we used a sign language handshape of the first letter of the desired room's color, and a spectrogram of a spoken command, "Go to the [color] room." b) In the Music domain, the agent's task is to play songs by selecting synthesized piano notes. For the cross domain goals, we used a spectrogram of a guitar that has played the desired song, and the sheet music for the song. The leftmost columns for Figures 4.2a and 4.2b represent two intra-domain goal representations for each task, while the remaining two are the cross-domain goal representations. The top and bottom rows for both domains are referred to as Task 1 and Task 2 in our experiments.

### 4.3.1 Maze Task

The first task we evaluate our approach on is a Maze, as shown in Fig 4.2a. Randomly generated mazes consist of 1-3 yellow, green, or blue colored rooms. The agent's actions are to move up, down, left or right, and its task is to navigate the maze. Goals designate the specific room the agent needs to reach. In order to specify the desired room using standard RL approaches, we would need to know its location or coordinates. Rather, we utilize two cross-domain goal representations: a sign language handshape indicating the first letter of the desired room color, and a spectrogram of a spoken command stating "Go to the [color] room." We use a dataset of sign language gestures from [19] for the handshape specification. For the speech specification, we recorded one sample of the spoken command for each colored room, and then varied the pitch to produce multiple samples.

We set the reward for the standard approach as $1$ if the agent is located in the desired room and $0$ otherwise. The agent's episode resets after $1000$ steps, as a terminal state requires knowledge of the goal location. During training, the agent could be initialized in any location on the map. During evaluation, it is initialized in the green room in the top maze in Figure 4.2a and in the yellow room in the bottom maze.

To train the reward network for this task, we used a momentum optimizer with an initial learning rate of $10^{-4}$ and momentum value of $.9$. We used a batch size of $32$. The embedding networks for the CDPR have the following architecture: Conv1 $\rightarrow$ maxpool $\rightarrow$ Conv2 $\rightarrow$ maxpool $\rightarrow$ FC1 $\rightarrow$ FC2, where Conv1 consists of 64 11x11 filters with stride 4, Conv2 consists of 192 5x5 filters with stride 2, FC1 outputs 400 features, and FC2 outputs 100. Each maxpool consists of a 3x3 filter with stride 2. Conv1, Conv2, and FC1 are each followed by batch normalization [20] and then ELU [21].

### 4.3.2 Music Playing Task

The next task we evaluate our approach on allows an agent to play synthesized piano notes by selecting keys and note durations. The agent can play 7 keys on a single scale of a-g, and each key can be a whole or half note. When the agent selects a note, a .wav file is generated that gets converted into a spectrogram, which represents its state. The task is for the agent to play a song, and the goals designate which song to play. In order to specify the desired song, we would need to know how to play the notes on the piano. Rather, we again use two cross-domain goal representations: a spectrogram of the desired song played on a synthesized guitar and the sheet music of the song, as shown in 4.2b. To obtain the samples of both specifications, we randomly generated notes and automatically created the sheet music and spectrograms for the guitar.

We set the reward for the standard approach as the total percentage of notes the agent has played correctly. The agent's episode resets after it plays notes for four complete bars. To avoid positive loops with the standard specification and CDPRs, the agent only receives a reward when it reaches this terminal state. During training and evaluation, the agent is initialized without any notes played.

To train the reward network for this task, we used an Adam optimizer [22] with an initial learning rate of $10^{-4}$. We used a batch size of $32$. The embedding networks for the CDPR have the following architecture: Conv1 $\rightarrow$ maxpool $\rightarrow$ Conv2 $\rightarrow$ maxpool $\rightarrow$ FC1, where Conv1 consists of 32 11x11 filters with stride 4, Conv2 consists of 64 5x5 filters with stride 2, and FC1 outputs 2048 features. Each maxpool consists of a 3x3 filter with stride 2. Conv1 and Conv2 are both followed by batch normalization and then ELU.

# CHAPTER 5

# RESULTS

Our experiments aimed to find if our reward network could effectively measure the distance between states in the agent's domain and cross-domain goals. To demonstrate the generality and accuracy of CDPRs, we compare against a standard hand-specified reward, and a state encoding reward. The state encoding reward learns features in the agent's domain and represents the goal as an intra-domain target image, and is represented by the negative euclidean distance between the agent's state and the intra-domain goal representation. For each of the tasks, the cross-domain goal representations were automatically generated.

## 5.1 Measuring Accuracy

### 5.1.1 Goal Retrieval Accuracy

We now describe a metric for measuring accuracy in reward functions. While evaluating with RL can suffice, it may be intractable to iterate over many goal instantiations. Therefore, we introduce the Goal Retrieval Accuracy (GRA) metric, which is motivated by image retrieval approaches. Given a set of states that were not seen during training time, we aim to find if the accurate cross-domain goal can be retrieved. In particular, this metric measures if the highest reward given to an unseen state matches the correct cross-domain goal. This measure gives an indication for how accurate the CDPRs are, and additionally indicates how well the learned rewards can generalize across multiple goal specifications.

### 5.1.2 Deep RL

We additionally measure how well the CDPRs work with RL. To further measure accuracy and generality, we used two unseen goal instantiations for each task, as shown in Figure 4.2.

We used Deep RL to solve the tasks with the architecture described in the paper [23]. We trained the network with an Adam optimizer with an initial learning rate of $10^{-4}$ and a batch size of $32$. To evaluate the agent's performance, we ran its learned policy on the task every $100$ episodes and measure a task-dependent evaluation accuracy.

## 5.2 Results

### 5.2.1 Goal Retrieval Accuracy

We randomly obtained $5000$ goal pairs that were not seen while training and obtained the GRA for each cross-domain goal representation. We also include results for random retrieval and the GRA for the state encoding, but these methods do not use cross-domain specifications.

| Reward Function | Hand shape | Speech | Guitar | Sheet Music |
|---|---|---|---|---|
| Random | 0.342 | 0.342 | 0.002 | 0.002 |
| State Encoding | 1.0 | 1.0 | 1.0 | 1.0 |
| CDPR | 0.98 | 0.986 | 0.904 | 0.888 |

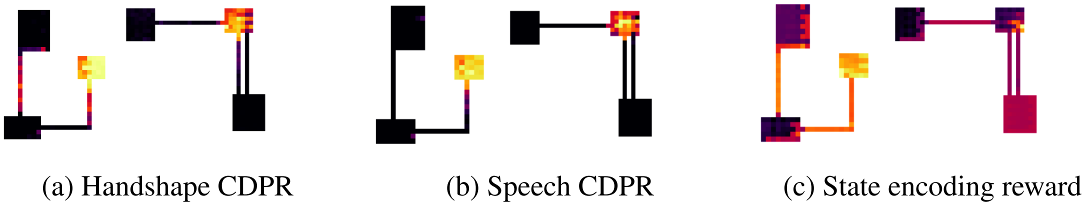
Table 5.1: Goal retrieval accuracy results. The handshape/speech and guitar/sheet music columns refer to the same results for the Maze and Music tasks, respectively.

### 5.2.2 Deep RL

We now report results for training the CDPRs for each task and running RL with the learned rewards.

*Maze Results*

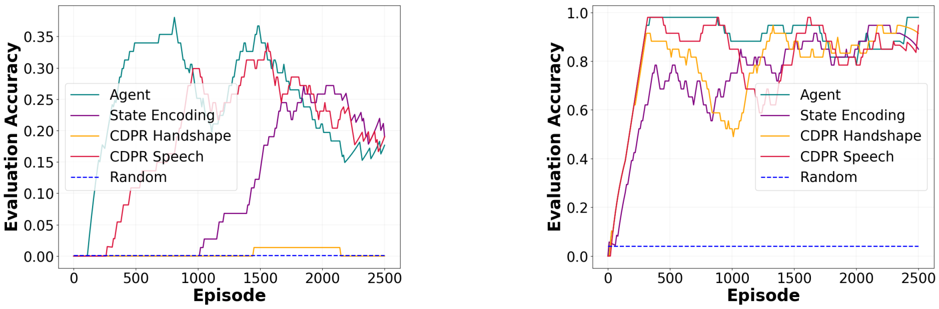
We first give qualitative results for the Maze task. Figure 5.1 shows a heat map of the rewards obtained from each location the agent could be spawned in. We also report the results for using the CDPRs with deep RL, as shown in Figure 5.2. We were interested in finding not only if the agent reached the goal, but also if it remained there, since this domain did not have terminal states. The desired goals were for the agent to reach the blue and green rooms, respectively, in the rooms displayed in Figure 4.2a.


(a) Handshape CDPR

(b) Speech CDPR

(c) State encoding reward

Figure 5.1: Qualitative results the Maze domain. To obtain these results, we spawned the agent in each location of the maze, and then produced a heat map of the rewards received.



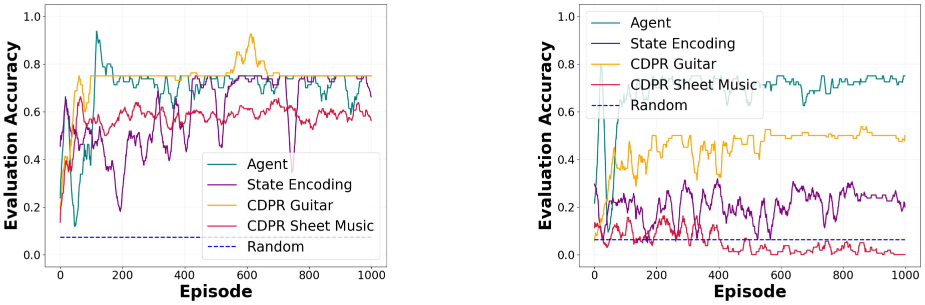
(a) Task 1

(b) Task 2

Figure 5.2: Smoothed results for running RL in the Maze domain. We ran RL with the goal specifications described in Figure 4.2a. The evaluation accuracy measures the percentage of time the agent spent in the correct room for each episode.

### *Music Results*

We now report results for the music domain in Figure 5.3. We do not have qualitative results for this task as the rewards are more difficult to visualize. The desired goals were for the agent to play the songs depicted in Figure 4.2b: {('a', W), ('b', W), ('b', H), ('c', H), ('d', W)} and {('f', W), ('a', H), ('e', H), ('a', H), ('c', H), ('e', W)}, where $W$ and $H$ are whole and half notes, respectively.



(a) Task 1

(b) Task 2

Figure 5.3: Smoothed results for running RL in the Music domain. We ran RL with the goal specifications described in Figure 4.2b. The evaluation accuracy measures the percentage of notes the agent played correctly at the end of each episode.

# CHAPTER 6
# DISCUSSION

## 6.1 Maze Results

Figure 5.1 shows a heat map of the rewards obtained from each location the agent could be spawned in. We do not show results for the standard reward, since this would just give rewards of 1 in the correct room and 0 otherwise. It is clear that CDPRs have learned to accurately reward the correct rooms for both the handshape and speech goal specifications. The largest reward values are given when the agent is in the correct room. We report the rewards for the feature encoding reward only for comparison, as these rewards are not directly learned. It is clear that the correct room will receive higher rewards than the rest, since the reward will be 0 when the state and goal are equal.

The next result we report is the GRA, shown in Table 5.1. Again, we show the encoding results for completeness, but the rewards will always yield a goal retrieval accuracy of 1. The handshape and speech CDPRs achieve high accuracy for retrieving the correct goal, demonstrating the accuracy of the approach and that it can generalize across new, unseen, goals.

Finally we evaluated the CDPRs using deep RL, as shown in Figure 5.2. The CDPR with the spoken command was able to perform well on both tasks. Interestingly, the CDPR with the handshape only performed well one one task. This may be surprising, since this goal representation achieved a high GRA.

It is clear that the GRA should not be the only metric for evaluating rewards. Rather, we should utilize multiple evaluation metrics, as incorrectly specified rewards have been known to lead to strange behavior. We have one explanation for why the handshape goal representation does not work for the first goal specification. If we again study the qualitative results from Figure 5.1, we observe that the handshape CDPR actually gives intermediate rewards in the hallways. This may lead to suboptimal policies, so even though the GRA is high, these intermediate rewards may lead to locally suboptimal policies. The feature encoding network also gives intermediate rewards, but these rewards will not introduce positive cycles since none of the rewards are greater than 0. Future work then should ensure we train on both goal images and random states obtained from the domain to ensure "false positives" such as this do not occur.

### 6.2 Music Results

The first result we report is the GRA, shown in Table 5.1. The guitar and sheet specifications also achieved high accuracy, but the results are clearly not as good as the Maze results. Just as in classification problems, we will need to determine how much error we are willing to tolerate to avoid providing solutions for every problem. Nevertheless, future work will aim to develop more sophisticated architectures that produce higher GRAs.

We use the RL results to further examine the correctness of the CDPRs, as shown in Figure 5.3. Both the guitar and sheet music goal specifications yielded results with high accuracy on the first song, each achieving accuracies of up to .75. The second song was slightly harder, as it had more notes. Both specifications achieved lower accuracy on the second song. In fact, the sheet music encoding learned a policy worse than random. The correspondence for this goal representation is likely more difficult than the guitar representation, which we see some evidence of in Figure 5.1. As we make improvements to the GRAs, we also expect the accuracy in RL to improve as well.

# CHAPTER 7

# CONCLUSIONS AND FUTURE WORK

We were able to form visual task descriptions for RL domains by utilizing the Reward Network. Our results indicate that we can accurately learn general reward functions specified through CDPRs. This shows that goals can be specified from domains that are different from an agent's own. An approach like this prevents users from getting bogged down in reward specification. Moreover, it makes reinforcement far more accessible, allowing it to be used in scenarios where the environment provided reward is non-existent or hard to discern. This has great implications in fields other than computer science as well, where reinforcement learning can be used to perform real world tasks after being trained on visual examples.

Future work would improve the accuracy of CDPRs and their performance on RL tasks. Currently, we require pairs to learn the correspondence between domains for CDPRs, and therefore, labeled data is required. A completely unsupervised approach to this would make CDPRs very effective and easy to use. Another interesting area to explore would be to learn policies that could generalize across goal specifications. Transfer learning is an area that would greatly benefit from an approach like this, and can be used to speed up training in multiple similar tasks.

In summary, we have shown how goals can be defined in alternative environments than the agents. We introduced Cross-Domain Perceptual Reward functions, and showed that our approach could achieve strong performance in tasks specified through cross-domain goal specifications.

# REFERENCES

[1] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. Cambridge Univ Press, 1998, vol. 1.

[2] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[3] A. Edwards, C. Isbell, and A. Takanishi, "Perceptual reward functions," *ArXiv preprint arXiv:1608.03824*, 2016.

[4] S. Levine, C. Finn, T. Darrell, and P. Abbeel, "End-to-end training of deep visuomotor policies," *ArXiv preprint arXiv:1504.00702*, 2015.

[5] P. Sermanet, K. Xu, and S. Levine, "Unsupervised perceptual rewards for imitation learning," *ArXiv preprint arXiv:1612.06699*, 2016.

[6] C. Finn, X. Y. Tan, Y. Duan, T. Darrell, S. Levine, and P. Abbeel, "Deep spatial autoencoders for visuomotor learning," *Reconstruction*, vol. 117, no. 117, p. 240, 2015.

[7] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and autonomous systems*, vol. 57, no. 5, pp. 469–483, 2009.

[8] P. Abbeel and A. Y. Ng, "Apprenticeship learning via inverse reinforcement learning," in *Proceedings of the twenty-first international conference on Machine learning*, ACM, 2004, p. 1.

[9] S. Schaal, "Is imitation learning the route to humanoid robots?" *Trends in cognitive sciences*, vol. 3, no. 6, pp. 233–242, 1999.

[10] F. Zhang, J. Leitner, M. Milford, B. Upcroft, and P. Corke, "Towards vision-based deep reinforcement learning for robotic motion control," *ArXiv preprint arXiv:1511.03791*, 2015.

[11] E. Tzeng, C. Devin, J. Hoffman, C. Finn, P. Abbeel, S. Levine, K. Saenko, and T. Darrell, "Adapting deep visuomotor representations with weak pairwise constraints," in *Workshop on the Algorithmic Foundations of Robotics (WAFR)*, 2016.

[12] A. A. Rusu, M. Vecerik, T. Rothörl, N. Heess, R. Pascanu, and R. Hadsell, “Sim-to-real robot learning from pixels with progressive nets,” *CoRR*, vol. abs/1610.04286, 2016.

[13] P. Isola, J. Zhu, T. Zhou, and A. A. Efros, “Image-to-image translation with conditional adversarial networks,” *CoRR*, vol. abs/1611.07004, 2016.

[14] A. B. L. Larsen, S. K. Sønderby, and O. Winther, “Autoencoding beyond pixels using a learned similarity metric,” *CoRR*, vol. abs/1512.09300, 2015.

[15] K. Bousmalis, G. Trigeorgis, N. Silberman, D. Krishnan, and D. Erhan, “Domain separation networks,” *CoRR*, vol. abs/1608.06019, 2016.

[16] S. Chopra, R. Hadsell, and Y. LeCun, “Learning a similarity metric discriminatively, with application to face verification,” in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, IEEE, vol. 1, 2005, pp. 539–546.

[17] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, “Image-to-image translation with conditional adversarial networks,” *ArXiv preprint arXiv:1611.07004*, 2016.

[18] D. Harwath, A. Torralba, and J. Glass, “Unsupervised learning of spoken language with visual context,” in *Advances in Neural Information Processing Systems*, 2016, pp. 1858–1866.

[19] A. L. C. Barczak, N. H. Reyes, M. Abastillas, A. Piccio, and T. Susnjak, “A new 2d static hand gesture colour image dataset for asl gestures,” *Research Letters in the Information and Mathematical Sciences*, vol. 15, pp. 12–20, 2011.

[20] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” *ArXiv preprint arXiv:1502.03167*, 2015.

[21] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, “Fast and accurate deep network learning by exponential linear units (elus),” *ArXiv preprint arXiv:1511.07289*, 2015.

[22] D. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *ArXiv preprint arXiv:1412.6980*, 2014.

[23] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, “Playing atari with deep reinforcement learning,” *ArXiv preprint arXiv:1312.5602*, 2013.