Inverse Reinforcement Learning for Team Sports: Valuing Actions and Players

by

Yudong Luo

B.Sc., Shanghai Jiao Tong University, 2018

Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of
Master of Science

in the
School of Computing Science
Faculty of Applied Sciences

Approval

Name:	**Yudong Luo**
Degree:	**Master of Science (Computing Science)**
Title:	**Inverse Reinforcement Learning for Team Sports: Valuing Actions and Players**
Examining Committee:	**Chair:** Parmit Chilana Assistant Professor
	Oliver Schulte Senior Supervisor Professor
	Hang Ma Supervisor Assistant Professor
	Anoop Sarkar Internal Examiner Professor School of Computing Science
Date Defended:	**June 15, 2020**

Abstract

A major task of sports analytics is to rank players based on the impact of their actions. Recent methods have applied reinforcement learning (RL) to assess the value of actions from a learned action value or Q-function. A fundamental challenge for estimating action values is that explicit reward signals (goals) are very sparse in many team sports, such as ice hockey and soccer. This paper combines Q-function learning with inverse reinforcement learning (IRL) to provide a novel player ranking method. We treat professional play as expert demonstrations for learning an implicit reward function. Our method alternates single-agent IRL to learn a reward function for multiple agents; we provide a theoretical justification for this procedure. Knowledge transfer is used to combine learned rewards and observed rewards from goals. Empirical evaluation, based on 4.5M play-by-play events in the National Hockey League (NHL), indicates that player ranking using the learned rewards achieves high correlations with standard success measures and temporal consistency throughout a season.

Keywords: Inverse Reinforcement Learning; Markov Games; Player Evaluation; National Hockey League

Dedication

To anyone who is interested in sports analytics and player ranking through reinforcement learning based method.

Acknowledgements

First and most important, I would like to express my sincere gratitude to my advisor Professor Oliver Schulte for the continuous support of my MSc study and research. All of his patience, motivation, enthusiasm, and immense knowledge helped me to finish this research work and my thesis. I could not have imagined having a better advisor for my MSc study at Simon Fraser University.

Besides my advisor, I would like to thank the rest of my thesis committee: Prof. Oliver Schulte, Prof. Hang Ma, Prof. Anoop Sarkar for their encouragement, insightful comments, and useful feedback.

My sincere thanks also goes to all of those with whom I have had the pleasure to work during this project and other related work, especially Guiliang, Mohan, Xiangyu, Yongjun, Jiaqi and Shuman. Each of them has provided me additional personal and professional guidance and I learned a lot about both scientific research and life in general from them.

Last but not least, nobody has been more important to me in the pursuit of this degree than my family members. I would like to thank my parents, whose love and guidance are always with me in whatever I pursue.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction and Overview

Sports analytics is a growing research field that provides professional methodology for analyzing sports data in order to facilitate decision making both before and during sports events. Over the past number of years, the advancements of data acquisition technique and the availability of powerful computing machine have led to the development of advanced sports specific algorithms. Statistics has been wildly utilized in sports analytics. Reinforcement learning (RL) is a promising method for sports analytics in recent years. Nowadays, sports analytics has become the intersection of several disciplines, including statistics, computer science, management, and health science.

There are many areas where sports analytics has been implemented. For instance, one of the major tasks is player evaluation, which supports drafting, coaching, and trading decisions and also helps to determine the best strategy for a team. Sports associations also use sports analysis to evaluate existing rules and the feasibility of introducing new rules to the game. Sports health mentors use it to evaluate players' mental and physical conditions.

In this thesis, we focus on the player evaluation task in sports analytics. Ranking players is becoming more and more important in sports industry. On the one hand, many companies (e.g. SportsLogiq, STATSports) and online platforms (e.g. nhl.com, whoscored.com) widely use player statistics to compare the performance of professional players. On the other hand, sports team managers and coaches are also desired to adopt sports analytic tools to monitor the quality of their players during matches as well as the entire season. Generally, ranking players requires researchers to define a relation of order between them with respect to some measure of their performance in the match. In turn, the most common approach is to quantify the impact of players' actions, and then aggregate action values for a player over the sequence of input matches [24, 13, 3].

Traditional player evaluation methods use statistics to evaluate action impact. For example, plus-minus (+/-) is a commonly used basic metric to measure the influence of player presence to the goals [15]. Some recent works modify the basic plus-minus metric by weighting the goals according to their importance, based on expected win probability, game time and game frequency [26], or with machine learning and survival models to estimate both expected goals and expected points to assess a player's over all defensive and offensive influence [10]. However, traditional sports evaluation metrics commonly face two major problems: 1) Most of the traditional player evaluation metrics focus only on the actions with immediate impact on goals, such as shots, but fail to consider the impact for actions that have significant long-term effects. This limitation is more severe for low-scoring games like ice hockey and soccer. 2) Traditional methods tend to assign fixed values to actions, regardless of the game context. A reasonable action impact algorithm should assign specific action value according to playing circumstances.

Several RL models have been proposed to tackle the above issues [23, 25, 13]. A Markov Game model was proposed in [23] to capture the game context for ice hockey and calculated a Q-value for each action. For each action, the Q-values estimate the probability that this action leads to the next goal, given the current game context. Deep reinforcement learning was adopted in [13] to train a deep Q-network. The difference between two consecutive Q-values is defined as the action impact. All these RL models use goals as the explicit reward signals, that is, 1 for scoring a goal and 0 for other actions. But the very sparse reward still presents two fundamental problems for Q-function learning: 1) Across game contexts, the Q-values show little variance, which means valuing actions with medium-term effects is still challenging using sparse reward. 2) Actions closely connected to goals are valued most highly and hence the performance evaluation is biased towards offensive players. To tackle the sparse reward issue, in this work, we propose a novel *inverse reinforcement learning method with domain knowledge* (IRL-DK) to recover a reward function for game dynamics.

In inverse reinforcement learning (IRL) [20], agents are assumed to act by optimizing an unobserved internal reward function. The learning task is to estimate the agents' rewards from their observed behavior (demonstrations). Sports are different from the general IRL settings, because some aspects of a player's reward can be inferred from domain knowledge. For instance, scoring a goal should have a relatively high reward because it helps the team to win a game. To benefit from both IRL and our domain knowledge, we introduce IRL-DK, which adopts knowledge transfer to combine the reward inferred from demonstrations and the one inferred from our domain knowledge. The final aggregated reward for a team is used to calculate a team Q-function. As in previous RL work, we use the team specific Q-function to calculate action impact value and rank players.

We model the team sports game as a Markov Game model ([12], see Chapter 3), where we treat home team (Home) and away team (Away) as two agents in the game. To learn team (agent) specific reward, we leverage single-agent IRL for multi-agent Markov Games through an *alternating learning* framework. Given observations of two teams A and B, we first treat team B as part of A's environment, then learn a reward function for team A in a single-agent Markov decision process (MDP). The procedure is repeated with the role of teams A and B reversed. We give a mathematical justification for this procedure in the sense that the single-agent MDP value function for one team agrees with its Markov Game value function. We apply alternation to generic Home and Away teams.

We apply our IRL-DK algorithm to the 2018-19 play-by-play data in the National Hockey League (NHL). The resulting distribution of top players is mixed among offensive and defensive players rather than concentrated among offensive players. Empirical comparison among 7 player evaluation metrics shows the high correlations with standard success measures and temporal consistency of our method.

Our main contributions may be summarized as follows.

1. A novel application of IRL to learning reward for teams in professional sports. Our method is general and can be applied to multi-agent dynamics in other domains.

2. A transfer learning method for combining sparse explicit rewards with learned dense implicit rewards.

3. An alternating learning procedure for leveraging single-agent IRL: For each agent in turn, the other agents are treated as part of the environment to define a single-agent MDP. We justify this procedure theoretically.

Finally, although our work is for sports analytics, the idea of transforming multi-agent Markov Game to a single agent MDP for evaluation problem and combining domain knowledge for reward learning is also useful for similar problem in other domain.

Thesis Outline. Apart from this Introduction and Overview chapter, there are six chapters left. Chapter 2 discusses the existing research work on action impact and player evaluation. Some heuristic idea and similar research work are also discussed in this chapter. Chapter 3 provides the general definition of Markov Game model and details its learning process. We give a theoretical justification for the transformation from multi-agent Markov Game to single-agent Markov Decision Process, which paves the way for our alternating IRL in Chapter 4. The second part of this chapter specifies the Markov Game model for ice hockey. The domain rules and the dataset we use are described. Chapter 4 gives the learning procedure of our inverse reinforcement learning with domain knowledge algorithm. We use alternating learning to recover team specific reward function for generic Home and Away teams. Chapter 5 assesses the quality of our recovered reward and policy induced from this reward function. Chapter 6 shows our downstream application player ranking and the empirical evaluation results. We summarize the thesis in Chapter 7 and discuss the future work.

Chapter 2

Background and Related Work

There exist different types of data for sports analytics, depending on which, different approaches can be applied. In summary, there are mainly three kind of data available. *Box score* data provide total action counts per player and match (e.g., number of goals scored). *Play-by-play* data, also known as event data, are logs of discrete action events specifying various properties of the action (e.g. action type, acting player, time and location. *Tracking* data record the location of each player at dense time intervals (e.g. for every broadcast video frame, or more frequently with stadium cameras). In this thesis, we focus on play-by-play data, and previous player ranking algorithms based on play-by-play data are discussed in this chapter.

2.1 Player Evaluation

Most approaches use the total value of a player's actions to rank players [1]. This reduces player evaluation to action evaluation problem.

+/- (Plus-Minus) is a commonly applied basic player evaluation metric using goals as the information only. It measures the influence of a player's presence on the goal scoring opportunity for his team. The original version of plus-minus assigns a positive one (+1) to a player if a goal is scored by the player's own team when this player is on the pitch, and assigns a negative one (-1) if the opposite team scores. Several works improved plus-minus with statistical techniques [5]. Some recent works modify the basic plus-minus metric, by weighting the goals according to their importance, based on expected win probability, game time and game frequency [26], or with machine learning and survival models to estimate both expected goals and expected points to assess a player's overall defensive and offensive influence [10].

Expected Goals assigns each shot the value of the probability that the shot leads to a goal and players are ranked by their total expected goals [16]. The similar method has been applied to passes recently, where the quality of a player's pass is quantified by the expected scoring probability [2]. A drawback of these ratings is that they evaluate only one type of action without modeling a player's overall performance.

Assessing expected impact for all actions, not only goals or shots, is challenging. One approach is to define a fixed look-ahead horizon, and train a classifier to predict whether an action will be followed by a goal within the horizon. For instance, the Total Hockey Rating (THoR) assesses the value of an action based on whether it is followed by a goal in the next 20 seconds [24]. The Valuing Actions by Estimating Probabilities (VAEP) method evaluates actions by whether the team scored or conceded a goal within the next k events (actions) [3]. The classifier could either be a traditional machine learning method like decision tree or a deep neural network.

Another approach to evaluating actions is quantifying the value-above-replacement. The most common value-above-replacement methods are Wins Above Replacement (WAR) and Goals Above Replacement (GAR). The contribution of a player is measured by estimating the difference of team's scoring/winning chances between the two situations where the target player is on the field compared to the target player is replaced by a replacement-level player. In this thesis, we take the replacement-level player to be a statistical league-average player. In other works, replacement-level represents a player of common skills available for minimum cost to a team.

2.2 Reinforcement Learning in Sports Analytics

State-of-the-art methods use RL for Q-function learning to assess the probability of scoring the next goal after a player's action.

Reinforcement Learning (RL) models event data of the form $s_0, a_0, r_0, s_1, s_1, r_1, \ldots, s_t, a_t, r_t$: an action a_t is chosen at state s_t, which results in a reward r_t and transition to a new state s_{t+1}. At next time stamp, another action a_{t+1} is chosen. In [23], a Markov model is applied to modeling ice hockey play-by-play data, where actions record the player movements and states capture the game context. They measured players performance by their expected Scoring Impact (SI). The expected scoring probabilities of player actions under different game context are modeled by a Q-function, and the Q-function represents the probability of scoring the next goal. The Markov model could be solved using dynamic programming based on the Bellman equation:

$$Q^{\pi}(s,a) = \sum_{s'} T(s'|s,a)[r(s,a) + \gamma \sum_{a'} \pi(s',a') Q^{\pi}(s',a')] \tag{2.1}$$

where $T(\cdot)$ gives the transition probability. $\pi(s,a)$ is a policy that specifies the probability of choosing action a at state s. This recurrence allows us to estimate the Q value at a current context given an estimate for the next Q values and transition probabilities T. The action impact value is defined as the advantage value:

$$\text{impact}(s,a) = Q(s,a) - V(s) \tag{2.2}$$

where $V(s)$ is calculated as

$$V^{\pi}(s) = \sum_{a} \pi(s,a)[r(s,a) + \gamma \sum_{s'} T(s,a,s')V^{\pi}(s')] \tag{2.3}$$

Instead of explicitly modeling transitions in an MDP, [13] adopted deep Q-network for Q function learning. Their model computes Q values to measure a player's expected probability of scoring the next goal with a temporal difference learning method called Sarsa [27]. The action impact is defined as the difference between two consecutive Q-values:

$$\text{impact}(s_t, a_t) = Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t) \tag{2.4}$$

2.3 Inverse Reinforcement Learning

Inverse reinforcement learning (IRL) is the problem of recovering the environment's reward function given observations of the behaviour of an optimally-behaving agent. For example, either the optimal policy π^* is known or we have the optimal trajectory sampled from unknown optimal policy. Once the reward function is recovered, we can then train an agent to find a policy based on that reward function, and as a result, the original behaviour can be replicated. IRL is slightly different from imitation learning. IRL recovers a reward function and learning a policy from the reward, rather than directly learning a policy from behaviour.

One advantage of IRL is that it does not require to directly interact with the environment, whereas in RL, it usually requires a virtual environment for agents to play with, in order to train a policy. We may also be able to solve the related problem of transfer learning, where the abstract goal the agent is trying to achieve is similar, but the specifics of the environment differ; a policy learned directly from another environment's optimal policy will probably not be successful in the new environment [11]. Another advantage is that the recovered reward can be used to explain the behaviour of an existing agent. For example, in [18], researchers would like to model the behaviour of bees. However, there is no real interest in recovering the behaviour of the bees, but instead in modelling their motivations.

2.4 Multi-agent Inverse Reinforcement Learning

Multi-agent IRL is the integration of multi-agent systems and (inverse) reinforcement learning. Consequently, multi-agent learning is challenging both technically and conceptually, and demands clear understanding of the problem to be resolved, the criteria for evaluation, and coherent research agendas. As a result, Multi-agent IRL is much less researched than single-agent IRL. [19] considered the task of learning a central controller to coordinate multiple agents. Our work addresses not a control problem, but a prediction problem, and there is no central controller, only agents acting independently of each other. [7] investigated a special case of multi agent, where only two agents play in the game and they share the same reward function. One agent is human and could teach the other agent, robot, to learn. [29] applied single-agent IRL to learn an individual reward function for World of Warcraft players. They do not incorporate explicit zero-sum rewards specified by the game rules. Also, they aim to model individual motivations, not to value actions and rank players. [31] presented a method for adversarial IRL that learns both a reward and a policy imitator represented as a GAN. Similarly, we assume the expert agents are following (approximately) a Nash equilibrium distribution: each agent is acting optimally given the observed policy of other agents. [31] further assumed the agents satisfy the conditions of a logistic quantal response equilibrium. Our focus is learning a reward function to complement the sparse goal signal, not on imitating policies. We work with a finite state-action space that does not require a GAN representation for policies. A novel aspect of our work not considered by [31] is combining learned rewards with observed rewards. Their MA-AIRL method was evaluated only on relatively small simulated environments, whereas our target application involves a complex real-world environment with a large amount of data. We leave for future work extending MA-AIRL with observed goals and scaling it to sports domains.

2.5 Inverse Reinforcement Learning and Knowledge Transfer

Mendez et al. [17] considered reward knowledge transfer among multiple tasks in an on-line setting. They aimed to accelerate the agent's ability of learning new tasks and reducing the amount of demonstrations required via continually building upon knowledge learned from previously demonstrated tasks. The lifelong function approximation is used to represent the reward functions for all tasks, which leads continual online transfer between the reward functions. In contrast, we consider knowledge transfer between two reward functions for the same task. [30] incorporated a known reward function using pretraining. The reward approximation network was first trained using supervised learning on human defined reward function (domain knowledge). Then the model was furthered updated using IRL algorithm. We also initialize our model with pre-trained parameters consistent with domain knowledge, but further use a Gaussian kernel regularization during training.

Chapter 3

Markov Games and Decision Processes

In this chapter, we first introduce the general Markov Game and then build Markov Game model for ice hockey. As our general formulation indicates, our approach can be used for sports in general, although this thesis focuses on ice hockey.

3.1 Markov Game Model

Markov games [12] extend MDPs to game theory [28]. So we first give a brief introduction of MDP.

Formally, an **MDP** [8] is defined by a state set S and action set A. A transition function $T : S \times A \to PD(S)$ defines the effect of actions over the state of the environment. (The notation $PD(X)$ denotes the set of probability distributions over a finite set X.) The reward $r : S \times A \to \mathbb{R}$ specifies the agent's task. Usually, there is a discounted factor $\gamma \in [0, 1)$ for agent's total return. A policy π is a distribution over actions given states: $\mathcal{S} \times \mathcal{A} \to [0, 1]$. Generally, the agent's objective is to find a policy mapping its interaction history to a current choice of action in order to maximize the expected sum of discounted reward

$$R_t = \mathbb{E}\{\sum_{j=0}^{\infty} \gamma^j r_{t+j}\} \tag{3.1}$$

where t is the current time stamp and r_{t+j} is the reward received j steps in the future. The discounted factor γ controls how much the future reward can affect the current optimal decision. The small value of γ focus on the near-term gain while large value of γ give large weight to future rewards. The value function for a MDP state is

$$V^{\pi}(s) = \sum_{a} \pi(a|s)[r(s, a) + \gamma \sum_{s'} T(s'|a, s)V^{\pi}(s')] \tag{3.2}$$

In its general form, a **Markov Game** [12] can be represented as a tuple $G = \langle \mathcal{S}, \boldsymbol{\mathcal{A}}, \boldsymbol{r}, \gamma, T \rangle$, where $\mathcal{S}$ is a finite set of states, $\boldsymbol{\mathcal{A}} = (\mathcal{A}_1, \ldots, \mathcal{A}_k)$ is a collection of finite action sets, one for each agent $1, \ldots, k$. For each agent, there is a real-valued reward function $\boldsymbol{r}_i : \mathcal{S} \times \mathcal{A}_i \to \mathbb{R}$, and a shared discount factor $0 < \gamma < 1$. The transition function $T : \mathcal{S} \times \boldsymbol{\mathcal{A}} \to PD(\mathcal{S})$ represents the environmental dynamics. An **MDP** is a single-agent Markov Game with $k = 1$. For each agent i, it attempts to maximize its expected sum of discounted rewards

$$R_{i,t} = \mathbb{E}\{\sum_{j=0}^{\infty} \gamma^j r_{i,t+j}\} \tag{3.3}$$

where $r_{i,t+j}$ is the reward received j steps into the future by agent i.

In Markov Game, a policy for agent i is a mapping $\pi_i : \mathcal{S} \to PD(\mathcal{A}_i)$. We assume the on-policy setting with a fixed policy vector $\pi_1, \ldots, \pi_k$. Note that since an agent's action probability is a function of the current game state, the agents' actions are independent of each other given the current game state. Focusing on a single agent i, we adopt game theory notation where $-i$ refers to the vector of the $k-1$ other agents. For instance, a policy vector can be decomposed as $\boldsymbol{\pi} = (\pi_i, \pi_{-i})$. Given a policy vector, a Markov Game defines a *game value function* for each agent i and state, which we denote by $G_i^{\pi_i, \pi_{-i}}(s)$. The game value represents the expected cumulative reward for agent i if the game starts in the state s, and satisfies the Bellman equation:

$$\begin{aligned} G_i^{\pi_i, \pi_{-i}}(s) = \sum_{a_i} \sum_{a_{-i}} \pi_i(a_i|s) \pi_{-i}(a_{-i}|s) \times \\ [\boldsymbol{r}_i(s, a_i, a_{-i}) + \gamma \sum_{s'} T(s'|a_i, a_{-i}, s) G_i^{\pi_i, \pi_{-i}}(s')], \end{aligned} \tag{3.4}$$

where $a_{-i} = (a_1, \ldots, a_{i-1}, a_{i+1}, \ldots, a_k)$ is a vector of actions by the agents other than i, and $\pi_{-i}(a_{-i}|s)$ is the probability of these independent actions given the policies of the agents other than i. This Bellman equation has a unique solution [27].

We now show that given a fixed policy vector π_{-i}, from agent i's perspective, a Markov Game $G = \langle \mathcal{S}, \boldsymbol{\mathcal{A}}, \boldsymbol{r}, \gamma, T \rangle$, is equivalent to a single-agent MDP. We define the **marginal MDP** as $M(\pi_{-i}) := \langle \mathcal{S}, \mathcal{A}_i, \boldsymbol{r}', \gamma, T' \rangle$, where

- $\boldsymbol{r}'(s, a_i) = \sum_{a_{-i}} r_i(s, a_i, a_{-i}) \cdot \pi_{-i}(a_{-i}|s)$
- $T'(s'|a_i, s) = \sum_{a_{-i}} T(s'|a_i, a_{-i}, s) \cdot \pi_{-i}(a_{-i}|s)$.

Proposition 1. *Given a Markov game G and policy vector π_{-i} for the agents other than i, the values of any policy π_i for agent i is the same in G and the marginal MDP $M(\pi_{-i})$:*

$$G_i^{\pi_i, \pi_{-i}}(s) = V^{\pi_i}(s) \tag{3.5}$$

Proof. We show that the Bellman equation for the marginal MDP is the same as for the Markov game. Since each Bellman equation has a unique value function as a solution, this implies that the value functions are the same.

$$\begin{aligned} V^{\pi_i}(s) &= \sum_{a_i} \pi_i(a_i|s)[r'(s,a_i) + \gamma \sum_{s'} T'(s'|a_i,s)V^{\pi_i}(s')] \\ &= \sum_{a_i} \pi_i(a_i|s)[\sum_{a_{-i}} r(s,a_i,a_{-i})\pi_{-i}(a_{-i}|s) \\ &\quad + \gamma \sum_{s'} \sum_{a_{-i}} T(s'|a_i,a_{-i},s)\pi_{-i}(a_{-i}|s)V^{\pi_i}(s')] \\ &= \sum_{a_i} \sum_{a_{-i}} r(s,a_i,a_{-i})\pi_i(a_i|s)\pi_{-i}(a_{-i}|s) \\ &\quad + \gamma \sum_{s'} \sum_{a_i} \sum_{a_{-i}} T(s'|a_i,a_{-i},s)\pi_i(a_i|s)\pi_{-i}(a_{-i}|s) \end{aligned} \tag{3.6}$$

The last equation agrees with the game Bellman equation (3.4).

This transformation is shown in Figure 3.1. To simplify but without loosing generalization, we shown a Markov Game with two agents A and B. By fixing B's policy and treating B as A's environment, the Markov Game can be transformed to a MDP, and the value function for agent A in MDP agrees with the value in Markov Game.

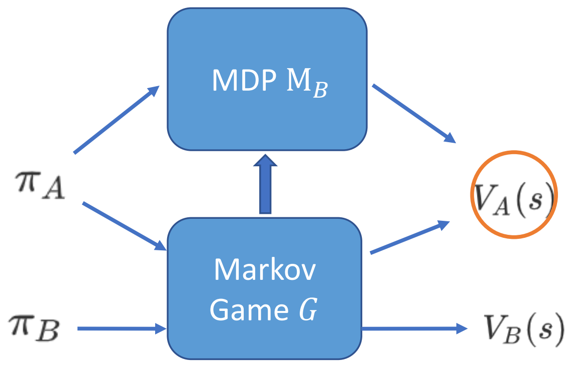

Figure 3.1: A and B refer to two generic players in a Markov Game. Transform Markov Game to an equivalent MDP, whose state values are the same for each player.

3.2 Markov Game Model for Ice Hockey

We summarize the rules of hockey and describe the how we build Markov Game model based on the data we gathered from NHL.

3.2.1 Domain Description: Hockey Rules and Dataset

A NHL game has 60 minutes of ice time, which are split into three periods, each 20 minutes in duration. There is a 15 minutes break between the periods. The hockey rink is divided into three zones by two blue lines. The zone between two blue lines is referred to as neutral zone. The zone with the team's own goal is called defensive zone and the zone that the team is trying to score in is the offensive zone, at the other end of the rink. In order to win the game, a team should score more goals than its opponent. If the score is tied at the end of regulation, there is a overtime, where first goal wins. If no goal is scored after the overtime during the regular season, the game moves into a shootout. During playoffs, overtime is repeated until a goal is scored.

As hockey being an intense sport, players are usually substituted on the fly after several minutes on the ice. In even strength situations of hockey game, there are six players in each side of the team, including five skaters and one goalie. The five skaters play different roles, including one Center, one Left Wing, one Right Wing, and two Defencemen. An infringement of rules will cause penalty to a team, which results in sending the offending player to a penalty box for two, four, or five minutes. This brings about manpower difference between two teams. The team being penalized is in shorthanded period while the opposing team is on the powerplay with a manpower advantage.

We utilize a play-by-play dataset constructed by SPORTLOGiQ company (proprietary, not available on line). This play-by-play data captures information of the NHL game from October 2018 to April 2019, which contains 4,534,017 events, covering 31 teams, 979 players and 1,202 games. A breakdown of this dataset is shown in Table 3.1. The data consists of game events around the puck, including the location and timestamp of a certain event, the identity of the player in possession and the action taken by this player, and other game context features (score difference, manpower, period, etc.). Table 3.2 shows an excerpt of data details. The X and Y coordinates are adjusted to the range [-100, 100] and [-42.5, 42.5] in feet, where the origin is center ice, the x-axis is along the length of the rink, and the y-axis is along the width as in Figure 3.2.

Number of Teams	31
Number of Players	979
Number of Games	1,202
Number of Events	4,534,017

Table 3.1: Size of the dataset

3.2.2 NHL Markov Game Setup

Our Markov Game model for ice hockey follows previous work [23]. We treat home team H and away team A as two agents in the game. At each timestamp, only one agent performs an action, and the agent not controlling the puck chooses no operation. To fit the Markov Game settings, at each state of the Markov Game, exactly one agent chooses No-op action. Each ice hockey game is modeled as a semi-episodic task [27], where games switch from

GID	PID	TID	Time	X	Y	GD	MP	OI	P	Z	Action	PO
329	274	11	16.7	38.5	-36.9	0	Even	5	1	OZ	Block	H
329	155	7	19.5	62.6	-38.4	0	Even	5	1	DZ	Lpr	A
329	212	11	38.4	-25.3	40.4	0	Even	5	1	DZ	Carry	H
329	1013	11	45.5	7.3	1.2	0	Even	5	1	NZ	Dumpin	H
329	273	11	51.4	38.5	-39.4	0	Even	5	1	OZ	Lpr	H

Table 3.2: An excerpt of NHL play-by-play dataset. GID=GameId, PID=PlayerId, TID=TeamId, GD=Goal Difference, MP=Manpower, OI=On Ice count, P=Period, Z=Zone, PO=Possession, OZ=Offensive Zone, DZ=Defensive Zone, NZ=Neutral Zone, H=Home, A=Away, H/A=Team who performs action, L=Location

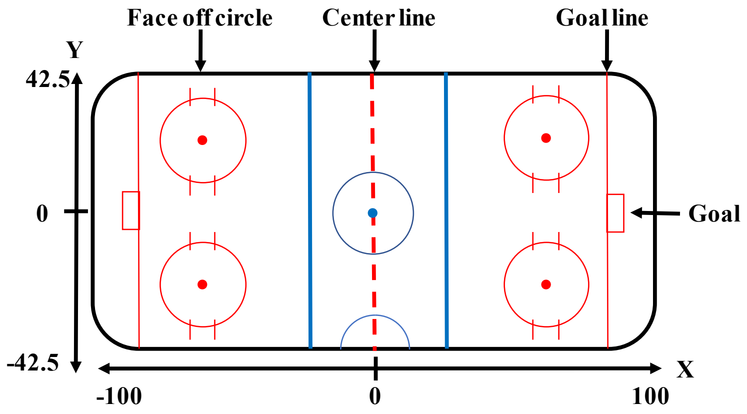

Figure 3.2: Ice hockey rink. Ice hockey is a fast-paced team sport, where two teams compete with each other and each team must shoot a puck into their opponent's net to score goals

episode to episode. Each episode starts either at the beginning of the game or right after a goal, and ends up with a goal or the end of the game. The transition function is calculated using the observed frequency $T(s,a,s') = p(s'|s,a) = O(s,a,s')/O(s,a)$, where $O(\cdot)$ counts the occurrence number in our dataset.

Feature	Notation	Range
Goal Difference	GD	[-8,8]
ManPower	MP	[-1,1]
Period	P	[1,3]
Team Identity	H/A	[0,1]
Location	L	[1,6]

Table 3.3: Range of features

Similar to previous Markov models for ice hockey [23, 25], we choose defining features for states, including game context, team identity (H/A) and location (L). A game context comprises Goal Difference (GD), ManPower (MP), and Period (P). GD is calculated as the number of home goals minus the number of away goals, ranging from -8 to 8. MP specifies shorthanded, even strength, and powerplay. P represents the current period, ranging from 1 to 3. (We do not consider overtime play.) We divide the hockey rink into 6 regions indexed by L based on the two blue lines to divide the X axis. The state features with their integer value range observed is summarized in Table 3.3. We add an absorbing goal state for each team, with no transition out of it. The dataset records 27 different action types, and home and away teams share the same action space.

Chapter 4

IRL with Domain Knowledge

In this chapter, We show our alternating learning procedure to leverage any single-agent IRL for multi-agent Markov Game based on proposition 1. We choose Maximum Entropy IRL because it provides an interpretable linear model for a reward function and scales to our large dataset. We first review the basic method, then discuss how to add domain knowledge to it.

4.1 Maximum Entropy IRL

The Maximum Entropy (MaxEnt) IRL built on Apprentice Learning (AL) [20] was introduced in [32] to solve the multiplicity issue of the possible reward functions. The basic idea is to match feature expectations between observed trajectories and optimal trajectories generated by learned policy based on recovered reward functions. Intuitively, if we compute the optimal policy for our recovered reward function (we assume the recovered reward is the same with or very similar to the true reward), we would expect that it generates the same trajectories as the optimal policy for the true reward function. Standard reinforcement learning methods are suitable for generating policy from recovered reward similar to the actual optimal policy.

In MaxEnt IRL, each state s is assigned a feature vector $\boldsymbol{f}_s \in \mathbb{R}^k$, and the reward function is parameterized as a linear function of a state with reward weights $\boldsymbol{\theta} \in \mathbb{R}^k$ as $r_{\boldsymbol{\theta}}(s) = \boldsymbol{\theta}^{\mathrm{T}} \boldsymbol{f}_s$. The state reward can be interpreted as the expected value over actions of the MDP reward $r(s, a)$. A trajectory $\zeta = (s_1, a_1, s_2, a_2, ..., a_{t-1}, s_t)$ is a sequence of state and action pair. The reward value for a trajectory ζ is simply the cumulative reward of visited states,

$$r(\zeta) = \sum_{s_j \in \zeta} \boldsymbol{\theta}^{\mathrm{T}} \boldsymbol{f}_{s_j} = \boldsymbol{\theta}^{\mathrm{T}} \boldsymbol{f}_{\zeta},$$

where $\boldsymbol{f}_{\zeta} = \sum_{s_j \in \zeta} \boldsymbol{f}_{s_j}$ is called the feature count of the trajectory. The observed agents' feature counts are calculated as $\tilde{\boldsymbol{f}} = \frac{1}{m} \sum_{\zeta} \boldsymbol{f}_{\zeta}$, where m is the number of trajectories. This observed feature count $\tilde{\boldsymbol{f}}$ summarizes the behavior of the expert demonstrator (agent). The intuition behind the MaxEnt IRL method is that observed feature count implicitly reveal the preferences of the expert, whom we can expect to manage to visit states they prefer more often. Therefore the objective of the learning procedure is to find a policy distribution P_π over trajectories whose expected feature counts match the expert's observed feature counts

$$\sum_{\zeta} P_\pi(\zeta)\boldsymbol{f}_\zeta = \tilde{\boldsymbol{f}}.$$

However, there are many possible policies that satisfy this constraint. Thus, MaxEnt IRL introduced a method to resolve this ambiguity and to discriminate between different policies based on the actual reward they accumulate, so that higher reward leads to higher probability of a trajectory.

Assume that agents act under a maximum entropy [9] policy, the probability of a demonstrated trajectory ζ increases exponentially with higher rewards. [32, Eq.4] shows that under mild assumptions, the exponential trajectory probability can be approximated by the expression

$$P(\zeta|\boldsymbol{\theta}, T) = \frac{e^{r_\zeta}}{Z(\boldsymbol{\theta}, T)} \prod_{s_{t+1}, a_t, s_t \in \zeta} P_T(s_{t+1}|a_t, s_t), \tag{4.1}$$

where $Z(\boldsymbol{\theta}, T)$ is the partition function and T is the state transition distribution. The policy π induced by this reward (parameterized by $\boldsymbol{\theta}$) is

$$\pi(a|s; \boldsymbol{\theta}) = \sum_{\zeta \in \sum_{s,a}} P(\zeta|\boldsymbol{\theta}), \tag{4.2}$$

where $\sum_{s,a}$ denotes the set of trajectories where action a is taken at state s.

Given the transition function T, MaxEnt IRL searches for the optimal $\boldsymbol{\theta}^*$ to maximize the log-likelihood $L(\boldsymbol{\theta})$ of the demonstrations

$$\boldsymbol{\theta}^* = \operatorname*{argmax}_{\boldsymbol{\theta}} L(\boldsymbol{\theta}) = \operatorname*{argmax}_{\boldsymbol{\theta}} \sum_{\zeta} \log P(\zeta|\boldsymbol{\theta}, T). \tag{4.3}$$

The maximum is obtained using gradient ascent; the gradient of $L(\boldsymbol{\theta})$ is the difference between observed and expected feature counts, which can be expressed in terms of state visitation frequencies D_s. The frequency of visiting a state given a policy can be computed with an iterative algorithm

$$\nabla L(\boldsymbol{\theta}) = \tilde{\boldsymbol{f}} - \sum_{\zeta} P(\zeta|\boldsymbol{\theta}, T)\boldsymbol{f}_\zeta = \tilde{\boldsymbol{f}} - \sum_{s_i} D_{s_i}\boldsymbol{f}_{s_i}. \tag{4.4}$$

4.2 MaxEnt IRL with Domain Knowledge

Directly using an IRL algorithm to recover the reward function from game dynamics models what situations professional players want to be in, that is, their internal reward function $\boldsymbol{r_\theta}$. But the MaxEnt approach fails to learn the importance of goals in a game, mainly because goals are such rare events in ice hockey. In addition, original MaxEnt IRL is for single-agent situation, while there are two agents (Home/Away team) in our ice hockey Markov Game. We first show how to transfer knowledge between the learned reward and our domain knowledge.

Previous RL methods define the reward function explicitly in terms of goals. The **rule reward function** $\boldsymbol{r}_K$ (for knowledge) assigns reward 1 for scoring a goal (i.e., getting the puck into the net) and 0 for other actions. Our knowledge transfer approach combines the MaxEnt likelihood function with the goal reward function through regularization:

$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, L(\boldsymbol{\theta}) + \lambda k(\boldsymbol{r_\theta}, \boldsymbol{r}_K), \tag{4.5}$$

where $\boldsymbol{r_\theta} = \boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{\psi}$, $\boldsymbol{r}_K = \boldsymbol{\theta}_K^{\mathrm{T}}\boldsymbol{\psi}$, $\boldsymbol{\psi} = [\boldsymbol{f}_{s_1}, ..., \boldsymbol{f}_{s_n}] \in \mathbb{R}^{k\times n}$ is the state feature matrix, λ is a trade-off parameter, and k is a kernel function that bridges the disparity between learned and knowledge reward functions. In this paper we use a Gaussian kernel $k(x_i, x_j) = \exp\{-||x_i - x_j||^2/2\}$. Following [30], we pre-train a parameter vector $\boldsymbol{\theta}_K$ to match our domain knowledge $\boldsymbol{r}_K$ and initialize $\boldsymbol{\theta}$ with $\boldsymbol{\theta}_K$. The gradient for $\boldsymbol{\theta}$ is given by

$$\nabla\boldsymbol{\theta} = \tilde{\boldsymbol{f}} - \sum_{s_i} D_{s_i}\boldsymbol{f}_{s_i} - \boldsymbol{\psi}[\lambda \exp(-\frac{1}{2}||\boldsymbol{r_\theta} - \boldsymbol{r}_K||^2) \circ (||\boldsymbol{r_\theta} - \boldsymbol{r}_K||)]^{\mathrm{T}} \tag{4.6}$$

This completes the description of our learning method. We next derive the regularizer (4.5) from a previous knowledge transfer framework.

Maximum Mean Discrepancy (MMD) [6] is an established framework for transferring knowledge between two distributions over random variables. Denote by X a random variable from distribution p, and by x the instantiations of X. A reproducing kernel Hilbert space (RKHS) $\mathcal{H}_k$ endowed by a kernel function $k(x, x')$ is a Hilbert space of functions $f(x) \to \mathbb{R}$ with inner product [6]. The element from this function space, $k(x, \cdot)$, satisfies the reproducing property: $\langle f(\cdot), k(x, \cdot)\rangle_{\mathcal{H}} = f(x)$. The $k(x, \cdot)$ is regarded as a feature map $\phi(x)$ where $k(x, x') = \langle \phi(x), \phi(x')\rangle_{\mathcal{H}}$. Similarly, denote by Y a random variable from distribution q, and by y the instantiations of Y. Formally, MMD defines the following difference measure

$$\begin{aligned} d_{\mathcal{H}_k}(p, q) &= \sup_{f\in\mathcal{H}_k} \mathbb{E}_x[f(x)] - \mathbb{E}_y[f(y)] \\ &= \sup_{f\in\mathcal{H}_k} \mathbb{E}_x[\langle \phi(x), f\rangle_{\mathcal{H}_k}] - \mathbb{E}_y[\langle \phi(y), f\rangle_{\mathcal{H}_k}] \end{aligned} \tag{4.7}$$

The mean embeddings of distribution p and q in $\mathcal{H}_k$, denoted by $\mu_k(p)$ and $\mu_k(q)$, satisfy $\mathbb{E}_x f(x) = \langle f(x), \mu_k(p)\rangle_{\mathcal{H}_k}$ and $\mathbb{E}_y f(y) = \langle f(y), \mu_k(q)\rangle_{\mathcal{H}_k}$ for all $f \in \mathcal{H}_k$. The MMD is expressed as the distance between two mean embeddings when function space in $\mathcal{H}_k$ is rich enough. The squared formula of MMD is defined as

$$d^2_{\mathcal{H}_k}(p, q) = ||\mu_k(p) - \mu_k(q)||^2 \tag{4.8}$$

In practice, given observations $\boldsymbol{x}$ of X and $\boldsymbol{y}$ of Y, an unbiased estimation of squared MMD is given by:

$$\begin{aligned}\hat{d}^2_{\mathcal{H}_k}(X, Y) = &\frac{1}{n_x^2}\sum_{i=1}^{n_x}\sum_{j=1}^{n_x} k(x_i, x_j) + \frac{1}{n_y^2}\sum_{i=1}^{n_y}\sum_{j=1}^{n_y} k(y_i, y_j) \\ &- \frac{2}{n_x n_y}\sum_{i=1}^{n_x}\sum_{j=1}^{n_y} k(x_i, y_j).\end{aligned} \tag{4.9}$$

Now we show how to get Equation (4.5). Since $\boldsymbol{\theta}$ is a function of sample, it denotes a random variable. As a result, $\boldsymbol{r_\theta}$ also defines a random variable, which we denote as $R_{\boldsymbol{\theta}}$ with observation $\boldsymbol{r_\theta}$. We also associate with $\boldsymbol{r}_K$ a constant random variable R_K with observation $\boldsymbol{r}_K$. The kernel function k is a Gaussian kernel in most knowledge transfer frameworks [14]. We want to maximize the log-likelihood of the observed trajectories as well as to minimize the MMD between two reward functions during training. The optimal $\boldsymbol{\theta}^*$ is derived by

$$\begin{aligned}\boldsymbol{\theta}^* &= \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, L(\boldsymbol{\theta}) - \alpha \hat{d}^2_{\mathcal{H}_k}(R_{\boldsymbol{\theta}}, R_K) \\ &= \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, L(\boldsymbol{\theta}) + 2\alpha k(\boldsymbol{r_\theta}, \boldsymbol{r}_K),\end{aligned} \tag{4.10}$$

where we have used the fact that the first two terms in Equation (4.9) are constant for a Gaussian kernel. Setting $\lambda = 2\alpha$ yields Equation (4.5).

Derivative for $\boldsymbol{\theta}$. Now we calculate derivative for $\boldsymbol{\theta}$ to get Equation (4.6).

$$\begin{aligned}\boldsymbol{\theta}^* &= \operatorname{argmax} L(\boldsymbol{\theta}) + \lambda k(\boldsymbol{r_\theta}, \boldsymbol{r}_K) \\ &= \operatorname{argmax} L(\boldsymbol{\theta}) + \lambda k(\boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{\psi}, \boldsymbol{\theta}_k^{\mathrm{T}}\boldsymbol{\psi}) \\ &= \operatorname{argmax} L(\boldsymbol{\theta}) + \lambda \exp(-||\boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{\psi} - \boldsymbol{\theta}_K^{\mathrm{T}}\boldsymbol{\psi}||^2/2) \\ \nabla\boldsymbol{\theta} &= \tilde{\boldsymbol{f}} - \sum_{s_i} D_{s_i}\boldsymbol{f}_{s_i} + \boldsymbol{\psi}[\lambda\exp(-\frac{1}{2}||\boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{\psi} - \boldsymbol{\theta}_K^{\mathrm{T}}\boldsymbol{\psi}||^2) \circ (-||\boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{\psi} - \boldsymbol{\theta}_K^{\mathrm{T}}\boldsymbol{\psi}||)]^{\mathrm{T}} \\ &= \tilde{\boldsymbol{f}} - \sum_{s_i} D_{s_i}\boldsymbol{f}_{s_i} - \boldsymbol{\psi}[\lambda\exp(-\frac{1}{2}||\boldsymbol{r_\theta} - \boldsymbol{r}_K||^2) \circ (||\boldsymbol{r_\theta} - \boldsymbol{r}_K||)]^{\mathrm{T}}\end{aligned} \tag{4.11}$$

4.3 Alternating MaxEnt IRL

We leverage single-agent MaxEnt IRL for multi-agent reward learning. As shown in Chapter 3, for a Markov Game with multiple agents, we can start from a certain agent's perspective and fix other agents' decisions as part of the environment, then the Markov Game can be transformed to a single-agent MDP with the state value for this agent being the same with Markov Game. In our ice hockey case, there are two teams Home and Away. We first treat Away team as part of Home's environment, then learn a reward function for Home team in a single-agent MDP. The procedure is repeated with the role of teams Home and Away reversed. This alternating learning procedure is shown in Figure 4.1.

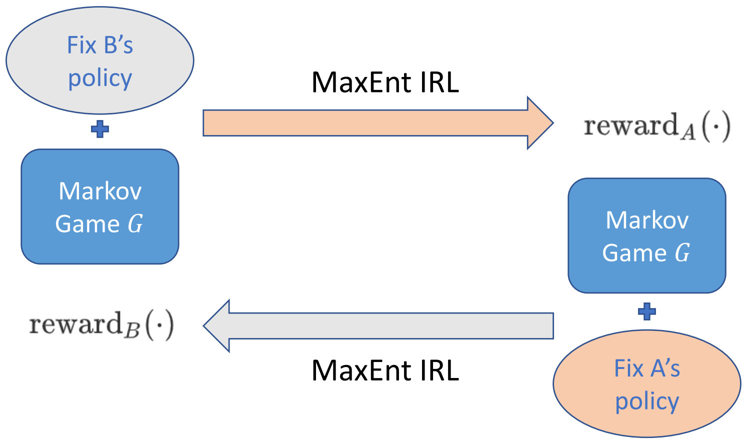

Figure 4.1: Alternating IRL for Markov Game to learn reward for different agents

Notice that our work uses IRL for *describing* agent behaviour, whereas most other IRL work has the *control* objective of building optimal agents. Previous work assumes that expert agents are following a Nash equilibrium distribution, which defines optimality in Markov Games [31]. Our optimality assumption is related but fundamentally different: Let $\hat{\pi}_A$,$\hat{\pi}_B$ be two policies for agents A and B estimated directly from the data that represent the agents' observed behaviour. Let $\hat{\boldsymbol{r}}_A$ and $\hat{\boldsymbol{r}}_B$ be two internal reward functions inferred from the data, where $\pi_A^{\boldsymbol{r}_A}$ and $\pi_B^{\boldsymbol{r}_B}$ are the *inferred* policies that optimize the agents' respective inferred reward functions. Our assumption is that *agents optimize against the observed policies of other agents* (i.e., $\hat{\pi}_A$ and $\hat{\pi}_B$ form an approximate Nash equilibrium). Previous control work computes policies such that agents optimize against the *inferred* optimal policies of other agents (i.e., $\pi_A^{\hat{\boldsymbol{r}}_A}$ and $\pi_B^{\hat{\boldsymbol{r}}_B}$ form an approximate Nash equilibrium). For describing a real-world domain like sports, our assumption is more realistic because i) teams have direct access only to the observed behavior of other teams, not to others' internal strategies ($\pi^{\hat{\boldsymbol{r}}}$), and ii) when an opponent's observed behavior $\hat{\pi}$ falls shorts of their optimal strategy $\pi^{\hat{\boldsymbol{r}}}$, successful teams take advantage of it.

Under the assumption of Nash equilibrium, we can transform the two agents Markov Game to a single agent MDP by fixing one agent's policy and treating that agent as another one's environment (cf. Chapter 3.1). Then the reward for different agents can be learned in an alternating approach.

Chapter 5

Evaluating the Learned Reward and Policy

In this chapter, we report different properties of the IRL-DK learned reward function from the ice hockey data.

5.1 Metrics

We first introduce the metrics used to evaluate the policy induced from the recovered reward. We compare the demonstrated trajectories with the probabilistic distribution over trajectories generated by our algorithm using two different metrics: negative log-likelihood (NLL) and modified Hausdorff Distance (MHD). the first is probabilistic and evaluates how likely the demonstrated trajectories are under the predicted distribution, and the second performs a more deterministic evaluation by estimating the spatial distances between the demonstrated trajectories and trajectories generated by our policy distribution.

The negative log-likelihood:

$$\mathrm{NLL}(\zeta) = -\log \prod_t P(s_{t+1}|s_t, a_t) \times \pi(a_t|s_t) \tag{5.1}$$

is the log-likelihood of a trajectory ζ under a policy $\pi(a|s)$. In our example, the trajectory ζ is the actual observed (demonstrated) trajectory. This metric measures the probability of drawing the demonstrated trajectory from the learned policy over all possible trajectories.

The modified Hausdorff Distance:

$$\begin{aligned} \mathrm{MHD}(\{\zeta_d\}, \{\zeta_g\}) &= \max(h(\{\zeta_d\}, \{\zeta_g\}), h(\{\zeta_g\}, \{\zeta_d\})) \\ h(\{\zeta\}, \{\hat{\zeta}\}) &= \frac{1}{|\{\zeta\}|} \sum_{\zeta_i \in \{\zeta\}} \min_{\hat{\zeta}_j \in \{\hat{\zeta}\}} ||\zeta_i - \hat{\zeta}_j|| \end{aligned} \tag{5.2}$$

is a physical measure of the distance between two set of trajectories. When the set size is one, it is actually the Euclidean distance. As in MaxEnt IRL, the choice of action has no effect on the total return of a trajectory, we represent the trajectory as a sequence of state when calculating HMD to simplify the problem. We sample the same number of trajectories generated by our policy as the number of demonstrated trajectory to compute the MHD.

5.2 Reward Density

Since our goal is to complement the sparse observed rewards with a dense reward signal that covers many situations, we would want the variance of learned rewards to be substantially higher than that of goal rewards. Recall that in previous RL-based methods, the reward function is very sparse as the reward is only defined on goals and ice hockey is a low-scoring game. Also, as a result of the sparse reward, the Q values from previous RL models show little variance among state and actions pairs.

Table 5.1 verifies that this is the case: the standard deviation (STD) of learned rewards is an order of magnitude higher, and the STD of the Q-function derived from the learned rewards is two orders of magnitude higher than that of the Q-function derived from goal rewards. The computation of the Q-values for IRL-DK is discussed in Section 6.1.1. For the goal reward function, we used the Q-values provided by [13], the state-of-the-art RL method for the goal reward.

Items	Mean	STD
Rule reward function (goals)	0.0000	0.0383
IRL-DK learned reward function	0.7964	0.1281
Q-values from goals (GIM)	0.4222	0.0963
Q-values from IRL-DK	5.1863	1.2207

Table 5.1: IRL-DK produces a dense reward signal with substantially higher variance than sparse explicit goal rewards.

Both our learned dense reward and induced dense Q values indicate that our method can capture more meaningful information from game context compared with using the rule based spares reward only.

We also shows the box plot in Figure 5.1 for rewards over states where the Goal Difference (GD) feature has different values. It is shown that states whose absolute GD (|GD|) value is smaller have larger reward. This consistent with the common sense that both agents in the game want to minimize the GD in order to win the game.

5.3 Policy Evaluation

To further evaluate how well the reward function recovered by our model rationalizes players' motivation and behavior, we solve the MDPs for the learned two reward functions to obtain two policies $\pi_H^{\boldsymbol{\theta}}$ and $\pi_A^{\boldsymbol{\theta}}$ for the home and away teams respectively. Then we compare the demonstrated trajectories with the probabilistic distribution over trajectories generated by the policies, using two common metrics: negative log-likelihood (NLL) and modified Hausdorff Distance (MHD) [30]. The definitions of NLL and MHD are introduced in Section 5.1.

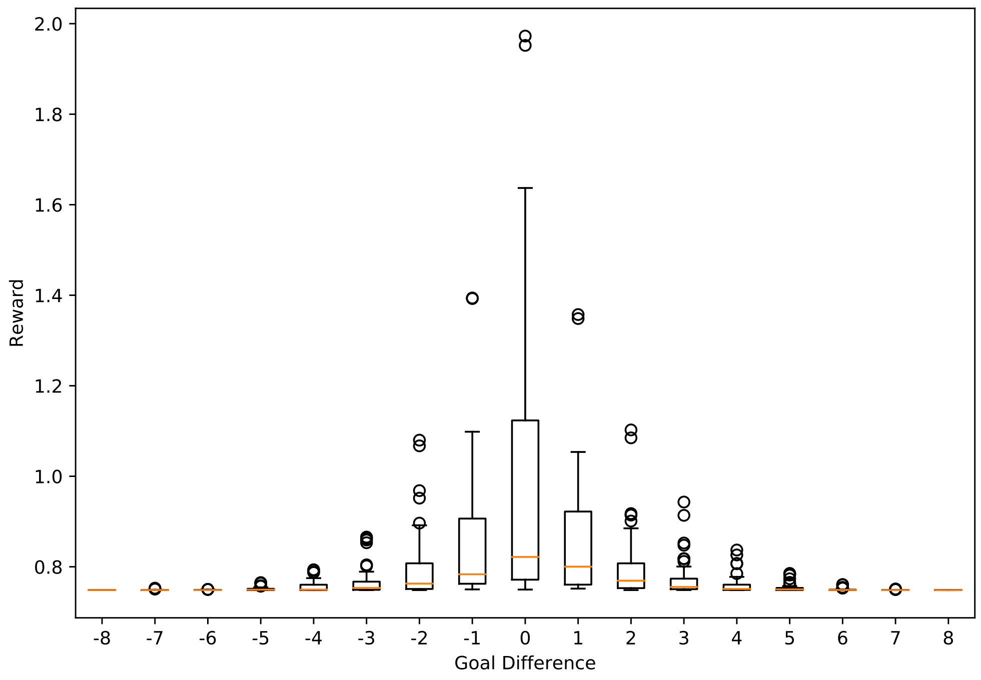

Figure 5.1: Rewards vary with game context

NLL calculates how likely the demonstrations are under policy π, and MHD is a spatial measure of the distance between demonstrated and generated trajectories. Table 5.2 shows the average results for both Home/Away teams. The policies optimal for the IRL reward with domain knowledge outperform their counterparts on both metrics.

Methods	NLL	HMD
Rule reward function (goals)	185.0	13.37
IRL learned reward function	53.9	9.71
IRL-DK learned reward function	**49.5**	**7.77**

Table 5.2: Evaluation of trajectory likelihoods under optimal policies derived from different reward functions. lower numbers indicate better approximations of expert behavior. For definitions see the text.

This experiment shows our policy induced by learned reward could reflect players' motivations more accurately. This also indicate that out recovered reward function is closer to the ground truth reward than the sparse one.

5.4 Learning Performance

To show the advantage of combining domain knowledge with MaxEnt IRL, we compare the learning process of MaxEnt IRL with and without domain knowledge during training. Figure 5.2 shows the average gradient changes for IRL and IRL-DK respectively.

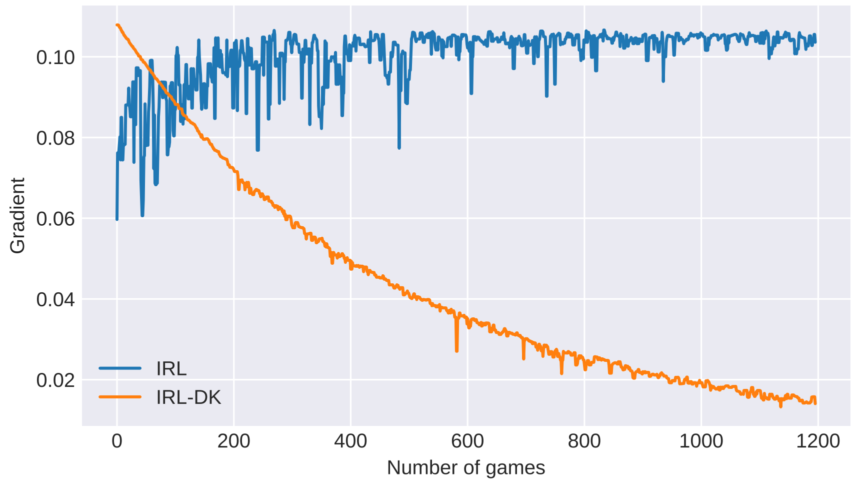

Figure 5.2: Average of gradients during training for IRL and IRL-DK

Both IRL and IRL-DK use the same Markov Game model described in Chapter 3 and the same learning rate. From the gradient changes, we can see that using MaxEnt IRL only is very unstable with oscillating gradients and fails to completely converge. Combining IRL with domain knowledge leads to a smoother training and faster convergence.

Chapter 6

NHL Player Ranking Experiments

In this chapter, we assess our Alternating IRL with domain knowledge in a downstream application, player ranking. Player ranking is one of the most important tasks in sports analytics. Players are evaluated by their observed performance over a set of games. Our player ranking approach is summarize in Figure 6.1. We first build a Markov Game model on our NHL dataset, then learn team specific reward functions for Home and Away team using our alternating learning framework. After recovering the reward, the MDPs could be solved to compute the value function and Q function. Finally, the action impact is defined and players are ranked by their total action impact values.

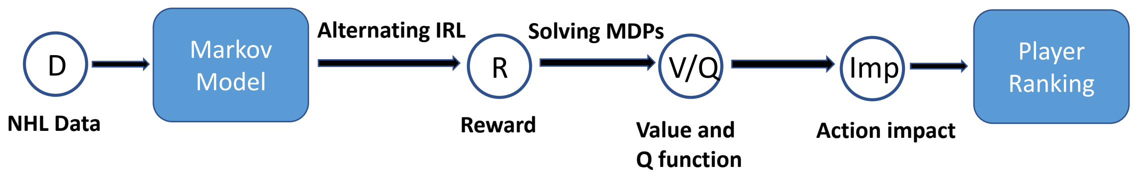

Figure 6.1: System Flow for Player Ranking

6.1 Player Ranking

We first define the action impact values and then give examples of player ranking.

6.1.1 Action Impact Values

Action impact, which quantifies the difference made by an action, has been used for player evaluation [23, 25, 13]. We adopt action impact values as a function of game context (Markov state) from [23]. For the home team H, the impact is defined by

$$\text{impact}_H(s,a) \equiv Q_{*H}^{\pi_H^{\boldsymbol{\theta}}}(s,a) - V_{*H}^{\pi_H^{\boldsymbol{\theta}}}(s), \tag{6.1}$$

where H is the team executing the action a, and the policy $\pi_H^{\boldsymbol{\theta}}$ is obtained by solving the single-agent MDP for the home team given the learned reward (cf. Section 5). Impact for the away team is defined similarly. Here, $V_*(\cdot)$ is calculated as

$$V_*(s) = \max_a [r(s,a) + \gamma \sum_{s'} T(s,a,s') V_*(s')] \tag{6.2}$$

After getting the optimal value function, the optimal $Q_*(\cdot)$ function is calculated as

$$Q_*(s,a) = r(s,a) + \gamma \sum_{s'} T(s,a,s') V_*(s') \tag{6.3}$$

This action impact function, commonly known as the advantage value in reinforcement learning, measures how much an action improves over the average action. The value of a state is defined as the expected total reward given a policy, and the Q-function and value function can be calculated using the Bellman equation [27], which has been discussed in Chapter 3.

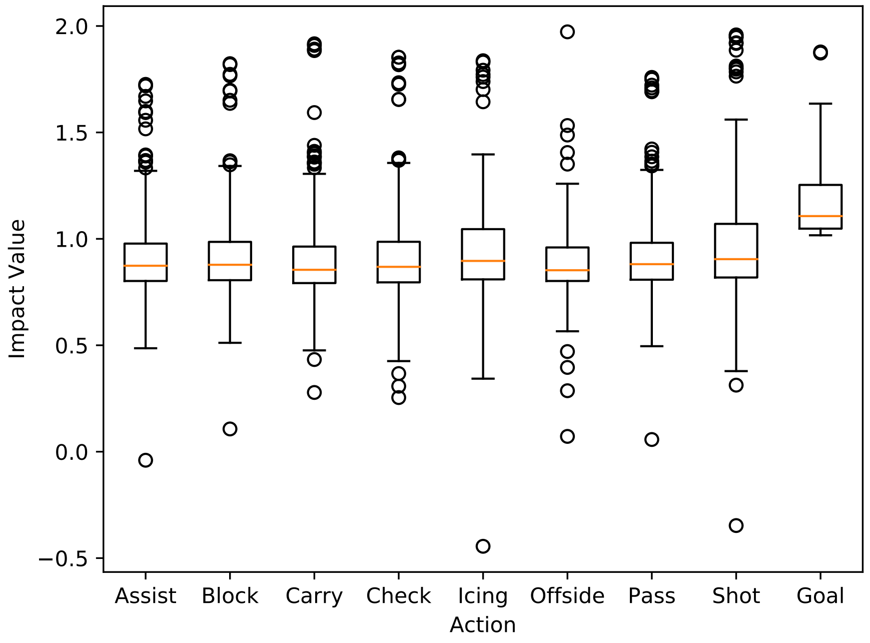

Figure 6.2: IRL-DK Action Impact Values vary with game context

Figure 6.2 shows a box plot for action impact values over all the states in the Markov Game. As the Q values used to calculate impact are relevant to the state frequency, to draw the box plot, we do not include the impact values whose state frequency is less than 0.05%. It is shown that the action impact values vary over states for different actions. The median value of goal impact is higher than that of other actions. However, as our method belongs to model-based RL, the probability of taking goal action is pretty low. It could be the case the Q value is similar to or smaller than state values for goal action, which leads to small or even negative goal action impact.

We compare the action impact with Goal Impact Metric (GIM). Figure 6.3 shows the box plot of GIM action impact for the same season. The medium impact value for most actions is near to zero and the impact of goal is much larger than that of other actions, which explains the top players given by GIM are most offensive players (Section 6.1.2), as offensive players score more goals than defensive players. We also notice that the medium impact value for shot is lower than other actions, which means GIM may fail to learn the importance of shot.

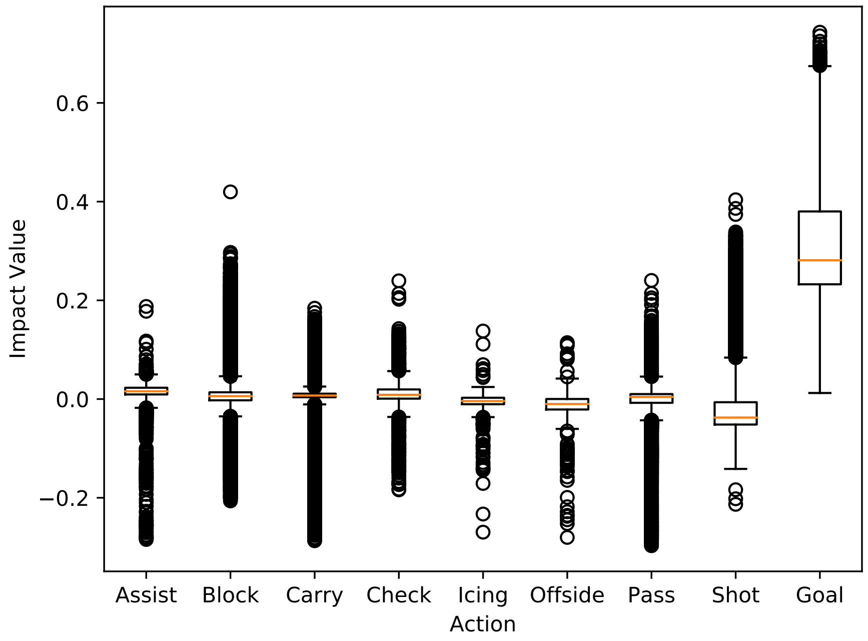

Figure 6.3: GIM Action Impact Values

6.1.2 Player Rankings

Following [13], the ranking score for a player is the sum of this player's total action impact values

$$\text{Score}_i = \sum_{s,a} n^i_{\mathcal{D}}(s,a) \times \text{impact}_{\text{team}_i}(s,a), \tag{6.4}$$

where $\mathcal{D}$ denotes our dataset, i is the playerId, $n^i_{\mathcal{D}}(s,a)$ is the occurrence number that player i performed action a at state s observed from $\mathcal{D}$, and team_i is the team of player i. The total impact is not normalized for time-on-ice (TOI), because TOI correlates with player strength. Dividing the ranking score by TOI therefore reduces the score differences among players. Note that impact values can be both positive and negative, so the total impact reflects the net value of a player's actions, rather than the total number of the actions.

Different from [23, 13] where all the players are evaluated together, we evaluate offensive players (Center, Left Wing, Right Wing) and defensive players (Defenceman, Goalie) separately with the following considerations. First, previous RL methods with sparse reward rank offensive players higher than defensive players in most cases. Second, these two types of players play different roles in a team under diverse strategies leading to distinct behavior.

Name	Assists	Goals	Points	Team	IRL Imp	Salary
Anze Kopitar	38	22	60	LA	14,407.5	11,000,000
Aleksander Barkov	61	35	96	FLA	14,386.6	6,900,000
Dylan Larkin	41	32	73	DET	14,018.2	7,000,000
Nathan Mackinnon	58	41	99	COL	13,836.6	6,750,000
Leon Draisaitl	55	50	105	EDM	13,739.9	9,000,000
Mark Scheifele	46	38	84	WPG	13,639.4	6,750,000
Jonthan Toews	46	35	81	CHI	13,357.3	9,800,000
Connor McDavid	75	41	116	EDM	13,309.6	14,000,000
Jack Eichel	54	28	82	BUF	13,288.8	10,000,000
Ryan O'Reilly	53	30	83	CAR	13,025.3	6,000,000

Table 6.1: 2018-19 Top-10 offensive players

Tables 6.1 and 6.2 list the top-10 highest impacts offensive and defensive players by our algorithm. All these players are NHL stars according to recent NHL 2019-20 top players news. Our ranking can be used to identify promising players. For instance, Miro Heiskane just began his career in 2017 and drew salaries below other top ranking players but is nominated as a top-50 defenceman by NHL [22]. *Our ranking does not have apparent bias towards offensive players* compared with two recent RL methods, Score Impact (SI) [23] and Goal Impact Metric (GIM) [13]. For instance, comparing the top-50 players, for the SI metric they are are all offensive players, for GIM all but one are offensive player, whereas our method contains 32 defencemen.

Name	Assists	Goals	Points	Team	IRL Imp	Salary
Drew Doughty	37	8	45	LA	15,229.4	12,000,000
Brent Burns	67	16	83	SJ	15,173.9	10,000,000
Roman Josi	41	15	56	NSH	14,356.7	4,000,000
John Carlson	57	13	70	WSH	14,279.2	12,000,000
Morgan Rielly	52	20	72	TOR	14,047.6	5,000,000
Ryan Suter	40	7	47	MIN	13,930.9	9,000,000
Mark Giordano	57	17	74	CGY	13,887.5	6,750,000
Duncan Keith	34	6	40	CHI	13,833.5	3,500,000
Erik Gustafsson	43	17	60	CHI	13,816.4	1,800,000
Miro Heiskane	21	12	33	DAL	13,678.3	925,000

Table 6.2: 2018-19 Top-10 defensive players

6.2 Empirical Evaluation

Similar to clustering problems, there is no ground truth for player evaluation. To assess player evaluation metrics, we follow previous work [23, 25, 13] and compute their correlation with commonly used statistic measurements like Assists, Goals, Points, as these statistics are generally regarded as important measures of a player's ability to impact a game.

We compare our method with the following player evaluation metrics. Plus-minus (**+/-**) is a commonly used basic metric to measure the influence of player presence to the goals [15]. Valuing Actions by Estimating Probabilities (**VAEP**) defines the impact of an action as its offensive score plus defensive score [3]. These two scores are defined as the differences between two consecutive scoring and conceding probabilities. Because our dataset was too large to be processed by the VAEP authors' code, we replaced the gradient-boosted tree of the original implementation by a neural network classifier. Win-Above-Replacement (**WAR**) estimates the difference of team's winning chance if a target player is replaced by an average player [4]. Expected Goal (**EG**) weights each shot by its chance of leading to a goal [16]. Scoring Impact (**SI**) is most related to our method, also based on a discrete Markov Game model but with the sparse goal reward [23, 25]. Goal Impact Metric (**GIM**) uses a deep Q-network with goal reward to predict Q-values and defines the difference between two consecutive Q-values as action impact [13]. We also adopt the maximum entropy **IRL** without domain knowledge as a baseline.

6.2.1 Season Totals: Correlations with Standard Success Measures

The following experiment computes the correlations with success measures over the entire 2018-19 season. The NHL official website (www.nhl.com/stats/player) provides 14 standard success measures, including Assists, Goals, Points, Game Play (GP), Game Wining Goal (GWG), Short-handed Goal (SHG), Power-play Goal (PPG), Shots (S), Short-handed Point (SHP), Power-play Point (PPP), Face-off Win Percentage (FOW), Points per game (P/GP), Shifts per game (SFT/GP), and Penalty Minute (PIM). The correlation is calculated over all the offensive or defensive players. We first generate players' total season values, then extract the corresponding success measures statistics for all the players. The correlation is computed between these two value lists, and the python function `numpy.corrcoef()` is utilized. The results for offensive and defensive players are shown in Tables 6.3 and 6.4.

Our method achieves the highest correlation in 10 out of 14 success measures except for goal and three goal related items (GWG, SHG, and PPG). For these measures, only SI shows a higher correlation, because it use the very sparse goal reward and is highly dominated by goal action. For GWG, our results are comparable to SI for both offensive and defensive player measures. For SHG and PPG, it achieves the second best results or comparable to the second best. The traditional plus-minus correlates poorly with all success measures. VAEP only achieves little correlation with success measures because their model is a classifier built on data with few positive labels and tends to assign similar impact value to all actions. EG is only the fourth best metric, because it only takes shots into account. IRL-DK achieves higher correlations than GIM, the most recent method, for every success measure except for

Methods	Assists	GP	Goals	GWG	SHG	PPG	S
+/-	0.269	0.086	0.282	0.278	0.118	0.124	0.156
VAEP	0.215	0.185	0.215	0.089	-0.074	0.160	0.239
WAR	0.591	0.322	0.742	0.571	0.179	0.610	0.576
EG	0.656	0.629	0.633	0.489	0.099	0.391	0.737
SI	0.717	0.633	**0.975**	**0.665**	**0.249**	**0.770**	0.860
GIM	0.757	0.772	0.781	0.518	0.147	0.477	0.795
IRL	0.855	0.872	0.812	0.587	0.123	0.513	0.901
IRL-DK	**0.882**	**0.887**	0.824	0.607	0.125	0.537	**0.907**

Methods	Points	SHP	PPP	FOW	P/GP	SFT/GP	PIM
+/-	0.285	0.179	0.157	0.012	0.306	0.109	0.100
VAEP	0.235	-0.076	0.185	0.021	0.204	0.129	0.172
WAR	0.692	0.147	0.605	0.040	0.699	0.396	0.145
EG	0.694	0.183	0.508	0.254	0.644	0.713	0.355
SI	0.869	0.204	0.708	0.135	0.728	0.639	0.361
GIM	0.818	0.151	0.561	0.289	0.705	0.751	0.372
IRL	0.891	0.207	0.696	0.294	0.741	0.818	0.437
IRL-DK	**0.908**	**0.213**	**0.734**	**0.298**	**0.769**	**0.820**	**0.446**

Table 6.3: Correlation with success measures (offensive). The line separates RL-based methods from others.

SHG. The difference is especially pronounced for defencemen and non-goal related measures (e.g. Points), due to GIM's goal bias.

6.2.2 Round-by-Round Correlations: Predicting Future Performance from Past Performance

A sport season normally consists of several rounds. A team or player will finish n competitions at the end of round n. We compute the correlation between player values at the end of round n and three main success measures, Assists, Goals, and Points, over the whole sport season. That is, we compute players' total values only based on the first n round data of the season. Then we extract the corresponding statistics (Assists, Goals, Points) for all the players and compute the correlation between these two value lists. Here, we also use the python function `numpy.corrcoef()`.

This experiment assesses the predictive power of different metrics, which allow us to infer the future performance of players. We also compute the *auto-correlation* for different metrics between players' round values and final season values. Auto-correlation evaluates the temporal consistency of a metric [21]. Since most players' strengths are stable throughout a season, a good player metric should show temporal consistency.

We focus on the four machine learning methods VAEP, SI, GIM, and IRL-DK. Figure 6.4 shows round-by-round correlation with Assists, Goals, Points, and the auto-correlation between round values and season total for offensive players. (Results for defensive players are

Methods	Assists	GP	Goals	GWG	SHG	PPG	S
+/-	0.173	0.132	0.144	0.177	0.235	-0.116	0.113
VAEP	0.054	-0.045	0.005	0.010	0.384	0.071	-0.016
WAR	0.204	0.028	0.365	0.275	0.097	0.246	0.186
EG	0.589	0.688	0.507	0.321	0.327	0.306	0.679
SI	0.607	0.488	**0.934**	**0.449**	**0.491**	**0.457**	0.709
GIM	0.702	0.862	0.596	0.263	0.130	0.170	0.764
IRL	0.809	0.941	0.686	0.415	0.268	0.347	0.908
IRL-DK	**0.852**	**0.959**	0.701	0.439	0.289	0.360	**0.920**

Methods	Points	SHP	PPP	FOW	P/GP	SFT/GP	PIM
+/-	0.175	0.107	-0.05	0.095	0.169	0.067	0.072
VAEP	0.042	0.065	-0.003	0.101	0.064	-0.036	-0.031
WAR	0.252	0.128	0.266	0.174	0.279	0.006	-0.089
EG	0.611	0.278	0.399	0.118	0.503	0.694	0.360
SI	0.720	0.174	0.488	0.103	0.521	0.499	0.272
GIM	0.730	0.085	0.358	0.140	0.471	0.706	0.438
IRL	0.841	0.281	0.549	0.182	0.557	0.776	0.549
IRL-DK	**0.865**	**0.307**	**0.571**	**0.185**	**0.574**	**0.778**	**0.570**

Table 6.4: Correlation with success measures (defensive)

similar.) IRL-DK is the *most stable model measured by auto-correlation*, and is the *best at predicting success measures*, even at the very beginning of the season.

To further evaluate the temporal consistency of our method, we also calculate the cross-season correlation between players ranking scores for three different seasons, following [21]. The model is trained on 19-20, 18-19, 17-18 NHL season separately and players are ranked based on their performances in these season games. The correlation result is shown in Table 6.5. The player rankings between two consecutive seasons correlate well with each other (above 0.75), which indicates that our rankings are temporally consistent even across seasons. Moreover, this means that our approach can be used to predict the future performance of a player in the next season from his performance in the current one.

Season	**19-20**	**18-19**	**17-18**
19-20	-	0.840	0.752
18-19	0.840	-	0.827
17-18	0.752	0.827	-

Table 6.5: IRL-DK cross-season player ranking correlation

6.3 Top players given by other metrics

Here we also show the top players given by other metrics, including SI, GIM, and VAEP.

Tables 6.6 and 6.7 list the top-10 highest impacts offensive and defensive players by SI. Tables 6.8 and 6.9 list the top-10 highest impacts offensive and defensive players by

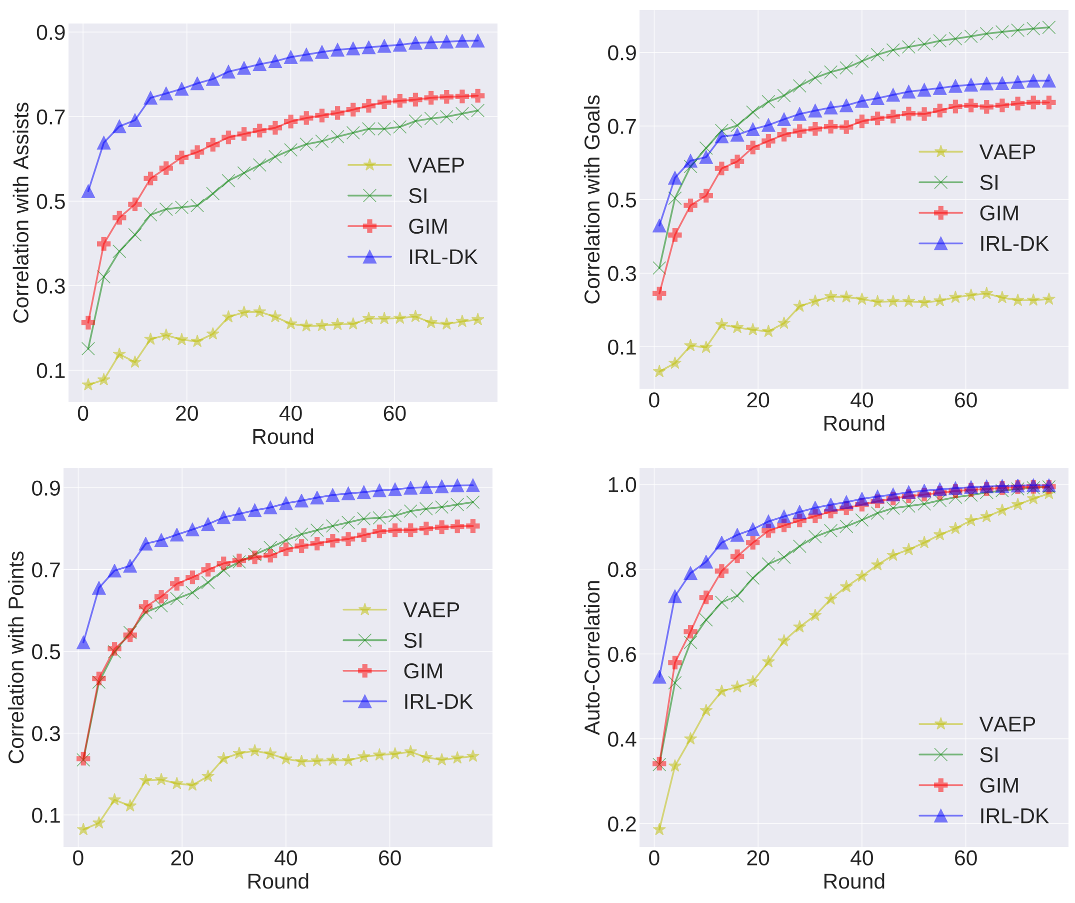

Figure 6.4: Correlations between round-by-round metrics and season totals for offensive players

GIM. The top-50 players given by VAEP [3] contains 38 offensive players and 12 defensive players. Tables 6.10 and 6.11 list the top-10 highest impacts offensive and defensive players by VAEP. All the rankings are based on the 2018-19 season.

Name	Assists	Goals	Points	Team	Salary
Alex Ovechkin	38	51	89	WSH	10,000,000
John Tavares	41	47	88	TOR	15,900,000
Leon Draisaitl	55	50	105	EDM	9,000,000
Cam Atkinson	28	41	69	CBJ	7,375,000
Alex Debrincat	35	41	76	CHI	800,000
Steven Stamkos	53	45	98	TBL	9,500,000
Jake Guentzel	36	40	76	PIT	7,000,000
Brayden Point	51	41	92	TBL	5,250,000
Patrick Kane	66	44	110	CHI	9,800,000
David Pastrnak	43	38	81	BOS	6,800,000

Table 6.6: SI Top-10 offensive players

Name	Assists	Goals	Points	Team	Salary
Morgan Rielly	52	20	72	TOR	5,000,000
Dougie Hamilton	21	18	39	CAR	6,000,000
Kris Letang	40	16	56	PIT	7,250,000
Mark Giordano	57	17	74	CGY	6,750,000
Jared Spurgeon	29	14	43	MIN	5,500,000
Matt Dumba	10	12	22	MIN	7,400,000
Shea Weber	19	14	33	MTL	6,000,000
Erik Gustafsson	43	17	60	CHI	1,800,000
Alex Pietrangelo	28	13	41	STL	7,500,000
Roman Josi	41	15	56	NSH	4,000,000

Table 6.7: SI Top-10 defensive players

Name	Assists	Goals	Points	Team	Salary
Sidney Crosby	65	35	100	PIT	9,000,000
Mark Scheifele	46	38	84	WPG	6,750,000
Leon Draisaitl	55	50	105	EDM	9,000,000
Jonathan Toews	46	35	81	CHI	9,800,000
Anze Kopitar	38	22	60	LA	11,000,000
Aleksander Barkov	61	35	96	FLA	6,900,000
John Tavares	41	47	88	TOR	15,900,000
Sean Couturier	43	33	76	PHI	4,500,000
Nicklas Backstrom	52	22	74	WSH	8,000,000
Connor McDavid	75	41	116	EDM	14,000,000

Table 6.8: GIM Top-10 offensive players

We refer to the ranking of the top 50 center, left wing, right wing, and defenceman given by NHL. The player names and rankings are shown in Appendix C.

All the top 10 offensive players from SI, GIM, and IRL-DK appear in the NHL rankings (including center, left wing, and right wing). 7 out of 10 top 10 offensive players from VAEP appears in NHL rankings. For defenceman, 8 out of 10 players by SI appear in NHL rankings. 7 out of 10 players by IRL-DK appear. Only 4 out of 10 players by GIM and VAEP appear

Name	Assists	Goals	Points	Team	Salary
Drew Doughty	37	8	45	LA	12,000,000
Jaccob Slavin	23	8	31	CAR	5,500,000
Samuel Girard	23	4	27	COL	700,000
T.J. Brodie	25	9	34	CGY	4,837,500
Michael Matheson	19	8	27	FLA	3,500,000
Thomas Chabot	41	14	55	OTT	832,500
Shea Theodore	25	12	37	VGK	5,200,000
Dmitry Orlov	26	3	29	WSH	6,500,000
Ivan Provorov	19	7	26	PHI	6,750,000
Morgan Rielly	52	20	72	TOR	5,000,000

Table 6.9: GIM Top-10 defensive players

Name	Assists	Goals	Points	Team	Salary
Jack Eichel	54	28	82	BUF	10,000,000
Ryan Getzlaf	34	14	48	ANA	8,275,000
Mika Zibanejad	44	30	74	NYR	5,350,000
Sidney Crosby	65	35	100	PIT	9,000,000
Brock Nelson	28	25	53	NYI	8,000,000
Lars Eller	23	13	36	WSH	4,000,000
Zach Aston-Reese	9	8	17	PIT	1,000,000
Chris Kreider	24	28	52	NYR	4,000,000
Nikita Kucherov	87	41	128	TBL	12,000,000
Leon Draisaitl	55	50	105	EDM	9,000,000

Table 6.10: VAEP Top-10 offensive players

in NHL rankings. While the NHL rankings do not provide a ground truth, the small overlap between the defensive rankings for GIM and VAEP with the NHL provides evidence that IRL-DK is more fair to defensive players. The SI defensive players also show good overlap but on an absolute scale, SI ranks defencemen much lower than forwards.

Name	Assists	Goals	Points	Team	Salary
Jonas Brodin	14	4	18	MIN	5,750,000
Jaccob Slavin	23	8	31	CAR	5,500,000
Mark Giordano	57	17	74	CGY	6,750,000
Jake Gardiner	27	3	30	TOR	3,650,000
Jordie Benn	17	5	22	NYR	2,400,000
Anton Stralman	15	2	17	TBL	5,500,000
Ryan Suter	40	7	47	MIN	9,000,000
Trevor Van Riemsdyk	11	3	14	CAR	2,500,000
Esa Lindell	21	11	32	PHI	7,000,000
Duncan Keith	34	6	40	ARI	3,500,000

Table 6.11: VAEP Top-10 defensive players

Chapter 7

Conclusion

We investigated multi-agent inverse reinforcement learning for professional ice hockey game analytics, a novel application area for AI. Our aim was to recover reward for complex game dynamics, which addresses the sparse reward issue for RL models. We introduced a transfer learning based regularization approach to incorporate domain knowledge in IRL. Based on the recovered reward function and calculated Q-values, we computed a context-aware player performance metric that provides a comprehensive evaluation for both offensive and defensive players in NHL by taking all their actions into account. In experiments our method shows no obvious bias for any player position, achieves highest correlation with most standard success measures among competing methods, and is most temporally consistent. While we have focused on ice hockey for concreteness, the IRL with domain knowledge method can be easily applied to a Markov Game model for any similar team sport. For future work, we can try to model the sports game as a partially observed Markov Game. Another important direction for future work is to learn reward functions at different levels, for instance, for individual teams and players.

Bibliography

[1] Jim Albert, Mark E Glickman, Tim B Swartz, and Ruud H Koning. *Handbook of Statistical Methods and Analyses in Sports.* 2017.

[2] Joel Brooks, Matthew Kerr, and John Guttag. Developing a data-driven player ranking in soccer using predictive model weights. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 49–55. ACM, 2016.

[3] Tom Decroos, Lotte Bransen, Jan Van Haaren, and Jesse Davis. Actions speak louder than goals: Valuing player actions in soccer. In *Proceedings of the 25th International Conference on Knowledge Discovery & Data Mining (KDD-19)*, pages 1851–1861, 2019.

[4] Tobias Gerstenberg, Tomer Ullman, Max Kleiman-Weiner, David Lagnado, and Josh Tenenbaum. Wins above replacement: Responsibility attributions as counterfactual replacements. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 36, 2014.

[5] Robert B Gramacy, Shane T Jensen, and Matt Taddy. Estimating player contribution in hockey with regularized logistic regression. *Journal of Quantitative Analysis in Sports*, 9(1):97–111, 2013.

[6] Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13(Mar):723–773, 2012.

[7] Dylan Hadfield-Menell, Stuart J Russell, Pieter Abbeel, and Anca Dragan. Cooperative inverse reinforcement learning. In *Proceedings of the 30th Conference on Neural Information Processing Systems (NIPS-16)*, pages 3909–3917, 2016.

[8] Ronald A Howard. Dynamic programming and markov processes. 1960.

[9] Edwin T Jaynes. Information theory and statistical mechanics. *Physical review*, 106(4):620, 1957.

[10] Tarak Kharrat, Ian G McHale, and Javier López Peña. Plus–minus player ratings for soccer. *European Journal of Operational Research*, 283(2):726–736, 2020.

[11] Sergey Levine, Zoran Popovic, and Vladlen Koltun. Nonlinear inverse reinforcement learning with gaussian processes. In *Proceedings of the 25th Conference on Neural Information Processing Systems (NIPS-11)*, pages 19–27, 2011.

[12] Michael L Littman. Markov games as a framework for multi-agent reinforcement learning. In *Proceedings of the 11st International Conference on Machine learning (ICML-94)*, volume 157, pages 157–163. 1994.

[13] Guiliang Liu and Oliver Schulte. Deep reinforcement learning in ice hockey for context-aware player evaluation. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI-18)*, page 3442–3448, 2018.

[14] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I Jordan. Deep transfer learning with joint adaptation networks. In *Proceedings of the 34th International Conference on Machine Learning (ICML-17)*, volume 70, pages 2208–2217, 2017.

[15] Brian Macdonald. An improved adjusted plus-minus statistic for nhl players. In *Proceedings of the 5th annual MIT Sloan Sports Analytics Conference*, volume 3, pages 1–8, 2011.

[16] Brian Macdonald. An expected goals model for evaluating nhl teams and players. In *Proceedings of the 6th annual MIT Sloan Sports Analytics Conference*, 2012.

[17] Jorge Armando Mendez, Shashank Shivkumar, and Eric Eaton. Lifelong inverse reinforcement learning. In *Proceedings of the 32nd Conference on Neural Information Processing Systems (NIPS-18)*, pages 4502–4513, 2018.

[18] P Read Montague, Peter Dayan, Christophe Person, and Terrence J Sejnowski. Bee foraging in uncertain environments using predictive hebbian learning. *Nature*, 377(6551):725–728, 1995.

[19] Sriraam Natarajan, Gautam Kunapuli, Kshitij Judah, Prasad Tadepalli, Kristian Kersting, and Jude Shavlik. Multi-agent inverse reinforcement learning. In *Proceedings of the 9th International Conference on Machine Learning and Applications*, pages 395–400, 2010.

[20] Andrew Y Ng, Stuart J Russell, et al. Algorithms for inverse reinforcement learning. In *Proceedings of the 17th International Conference on Machine Learning (ICML-00)*, volume 1, page 2, 2000.

[21] Stephen Pettigrew. Assessing the offensive productivity of nhl players using in-game win probabilities. In *Proceedings of the 9th annual MIT sloan sports analytics conference*, volume 2, page 8, 2015.

[22] Rob Reese. Fantasy defenseman top 50 rankings for 2019-20. `https://www.nhl.com/news/nhl-fantasy-hockey-top-50-defenseman-rankings-2019-20/c-282830728`, 2019. [Online; accessed 15-October-2019].

[23] Kurt Routley and Oliver Schulte. A markov game model for valuing player actions in ice hockey. In *Proceedings of the 31st Uncertainty in Artificial Intelligence (UAI-15)*, pages 782–791, 2015.

[24] Michael Schuckers and James Curro. Total hockey rating (thor): A comprehensive statistical rating of national hockey league forwards and defensemen based upon all on-ice events. In *Proceedings of the 7th annual MIT sloan sports analytics conference*, 2013.

[25] Oliver Schulte, Mahmoud Khademi, Sajjad Gholami, Zeyu Zhao, Mehrsan Javan, and Philippe Desaulniers. A markov game model for valuing actions, locations, and team performance in ice hockey. *Data Mining and Knowledge Discovery*, 31(6):1735–1757, 2017.

[26] Steven R Schultze and Christian-Mathias Wellbrock. A weighted plus/minus metric for individual soccer player performance. *Journal of Sports Analytics*, 4(2):121–131, 2018.

[27] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction.* 1998.

[28] John Von Neumann and Oskar Morgenstern. Theory of games and economic behavior, 2nd rev. 1947.

[29] Baoxiang Wang, Tongfang Sun, and Xianjun Sam Zheng. Beyond winning and losing: Modeling human motivations and behaviors with vector-valued inverse reinforcement learning. In *Proceedings of the 15th AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE-19)*, volume 15, pages 195–201, 2019.

[30] Markus Wulfmeier, Dushyant Rao, and Ingmar Posner. Incorporating human domain knowledge into large scale cost function learning. *arXiv preprint arXiv:1612.04318*, 2016.

[31] Lantao Yu, Jiaming Song, and Stefano Ermon. Multi-agent adversarial inverse reinforcement learning. In *Proceedings of the 36th International Conference on Machine-Learning (ICML-19)*, pages 7194–7201, 2019.

[32] Brian D Ziebart, Andrew Maas, J Andrew Bagnell, and Anind K Dey. Maximum entropy inverse reinforcement learning. In *Proceedings of the 23rd AAAI Conference on Artificial Intelligence (AAAI-08)*, 2008.

Appendix A

Code

Our main code can be retrieved from github. https://github.com/miyunluo/IRL-icehockey

Appendix B

Box plot

Figure B.1 shows a box plot for action Q values over all the states given by our method (**IRL-DK**). The Q value for goal action is lower than other actions because our model is a model-based method. Goal is a rare action appears in our dateset and can only lead to the end state in our Markov Game model. The Q values here do not represent the importance of actions.

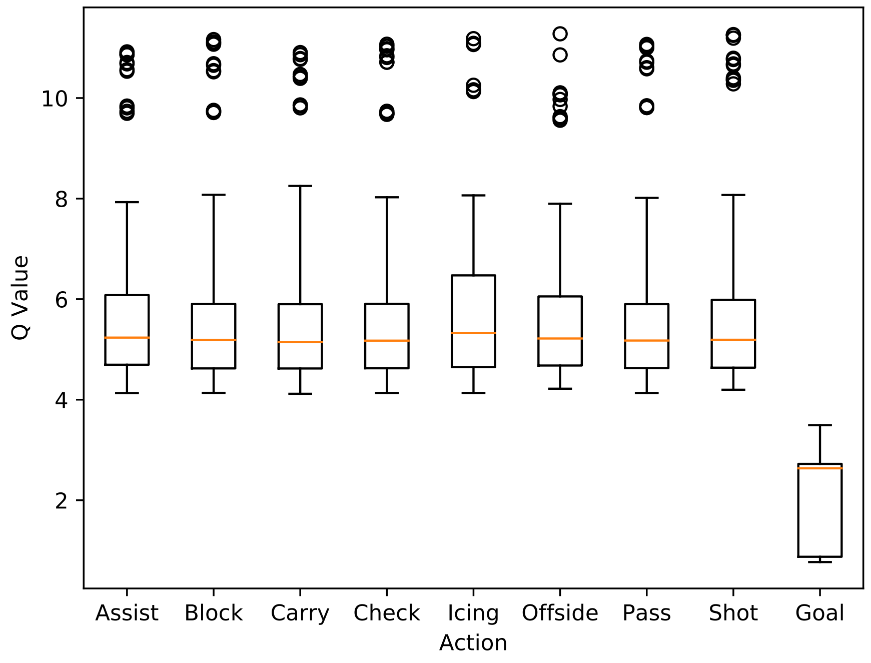

Figure B.1: IRL-DK Action Q Values

We also give the box plot for Q values from **GIM** shows in Figure B.2.

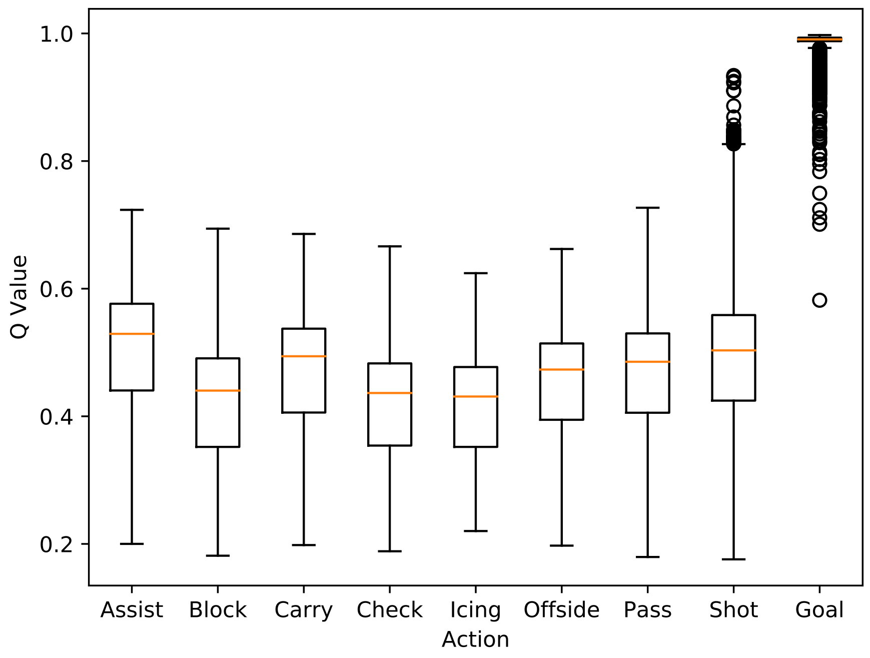

Figure B.2: GIM Action Q Values

Appendix C

19-20 Top 50 rankings by NHL

This ranking information is obtained from NHL official website. C represents center, LW and RW for left wing and right wing.

Fantasy center top 50 rankings for 2019-20 includes: 1. Connor McDavid, C, 2. Nathan MacKinnon, C, 3. Sidney Crosby, C, 4. Auston Matthews, C, 5. Aleksander Barkov, C, 6. Tyler Seguin, C, 7. Steven Stamkos, C, 8. John Tavares, C, 9. Elias Pettersson, C, 10. Evgeni Malkin, C, 11. Patrice Bergeron, C, 12. Mark Scheifele, C, 13. Sebastian Aho, C, 14. Jack Eichel, C, 15. Brayden Point, C, 16. Evgeny Kuznetsov, C, 17. Nicklas Backstrom, C, 18. Sean Monahan, C, 19. Mika Zibanejad, C, 20. Tomas Hertl, C/LW, 21. Logan Couture, C, 22. Matt Duchene, C, 23. Dylan Larkin, C, 24. Mathew Barzal, C, 25. Sean Couturier, C, 26. Jonathan Toews, C, 27. Ryan O'Reilly, C, 28. Jack Hughes, C, 29. Ryan Nugent-Hopkins, C/LW, 30. Brayden Schenn, C/LW, 31. Max Domi, C/LW, 32. Dylan Strome, C, 33. Anze Kopitar, C, 34. Roope Hintz, C/LW, 35. Nico Hischier, C, 36. Paul Stastny, C, 37. Bo Horvat, C, 38. Vincent Trocheck, C, 39. Nazem Kadri, C, 40. Pierre-Luc Dubois, C, 41. William Karlsson, C, 42. Ryan Johansen, C, 43. Nick Schmaltz, C/LW, 44. Ryan Getzlaf, C, 45. David Krejci, C, 46. Andreas Athanasiou, C/LW, 47. Anthony Cirelli, C, 48. Cody Glass, C, 49. Eric Staal, C, 50. Kevin Hayes, C.

Fantasy left wing top 50 rankings for 2019-20 includes : 1. Alex Ovechkin, LW, 2. Brad Marchand, LW, 3. Leon Draisaitl, C/LW, 4. Johnny Gaudreau, LW, 5. Artemi Panarin, LW, 6. Taylor Hall, LW, 7. Claude Giroux, C/LW/RW, 8. Gabriel Landeskog, C/LW, 9. Alex DeBrincat, LW/RW, 10. Jake Guentzel, LW/RW, 11. Matthew Tkachuk, LW, 12. Jonathan Huberdeau, LW, 13. Filip Forsberg, LW, 14. Jamie Benn, C/LW, 15. Timo Meier, LW/RW, 16. Max Pacioretty, LW, 17. Jonathan Marchessault, C/LW, 18. Teuvo Teravainen, LW/RW, 19. Clayton Keller, LW/RW, 20. Kyle Connor, LW, 21. Jeff Skinner, C/LW, 22. Mike Hoffman, LW/RW, 23. Evander Kane, LW, 24. Brady Tkachuk, LW, 25. Rickard Rakell, LW/RW, 26. James van Riemsdyk, LW, 27. Nikolaj Ehlers, LW/RW, 28. Nikita Gusev, LW, 29. Alex Galchenyuk, C/LW, 30. J.T. Miller, LW/RW, 31. Mikael Granlund, LW/RW, 32. Chris Kreider, LW, 33. Jaden Schwartz, LW, 34. Anders Lee, LW, 35. Nino Niederreiter, LW/RW, 36. Jake DeBrusk, LW/RW, 37. Zach Parise, LW, 38. Jakub Vrana, LW, 39. Andreas Johnsson, LW/RW, 40. Tyler Bertuzzi, LW/RW, 41. Max Comtois, LW, 42. Victor Olofsson, LW/RW, 43. Gustav Nyquist, LW/RW, 44. Marcus Johansson, LW,

45. Tomas Tatar, LW/RW, 46. Jason Zucker, LW/RW, 47. Jonathan Drouin, C/LW, 48. Alexandre Texier, C/LW, 49. Andre Burakovsky, LW/RW, 50. Ondrej Palat, LW.

Fantasy right wing top 50 rankings for 2019-20 include: 1. Nikita Kucherov, RW, 2. Patrick Kane, RW, 3. David Pastrnak, RW, 4. Mitchell Marner, RW, 5. Mikko Rantanen, RW, 6. Blake Wheeler, RW, 7. Vladimir Tarasenko, RW, 8. Patrik Laine, LW/RW, 9. Mark Stone, RW, 10. Alexander Radulov, RW, 11. Phil Kessel, RW, 12. Brock Boeser, RW, 13. Joe Pavelski, C/RW, 14. Jakub Voracek, RW, 15. Viktor Arvidsson, LW/RW, 16. William Nylander, C/RW, 17. Evgenii Dadonov, LW/RW, 18. Kaapo Kakko, RW, 19. Tom Wilson, RW, 20. Elias Lindholm, C/RW, 21. Cam Atkinson, RW, 22. Brendan Gallagher, RW, 23. Kyle Palmieri, RW, 24. T.J. Oshie, RW, 25. Andrei Svechnikov, LW/RW, 26. Anthony Mantha, LW/RW, 27. Sam Reinhart, C/RW, 28. Jordan Eberle, RW, 29. Mats Zuccarello, RW, 30. Reilly Smith, RW, 31. Patric Hornqvist, RW, 32. Dustin Brown, RW, 33. Kevin Labanc, LW/RW, 34. Kasperi Kapanen, RW, 35. David Perron, LW/RW, 36. Ondrej Kase, RW, 37. Pavel Buchnevich, RW, 38. Ryan Donato, LW/RW, 39. Yanni Gourde, LW/RW, 40. Josh Anderson, RW, 41. Travis Konecny, RW, 42. Tyler Johnson, LW/RW, 43. Jakob Silfverberg, RW, 44. Wayne Simmonds, RW, 45. James Neal, LW/RW, 46. Tyler Toffoli, RW, 47. Drake Batherson, C/RW, 48. Robert Thomas, C/RW, 49. Josh Bailey, LW/RW, 50. Dominik Kahun, LW/RW.

Fantasy defenseman top 50 rankings for 2019-20 includes: 1. John Carlson, 2. Victor Hedman, 3. Cale Makar, 4. Roman Josi, 5. Alex Pietrangelo, 6. Torey Krug, 7. Kris Letang, 8. Dougie Hamilton, 9. Shea Theodore, 10. Quinn Hughes, 11. Morgan Rielly, 12. Zach Werenski, 13. Seth Jones, 14. Tyson Barrie, 15. Ivan Provorov, 16. Charlie McAvoy, 17. John Klingberg, 18. Neal Pionk, 19. Miro Heiskanen, 20. Tony DeAngelo, 21. Ryan Ellis, 22. Mark Giordano, 23. Oscar Klefbom, 24. Keith Yandle, 25. Shea Weber, 26. Mikhail Sergachev, 27. Ryan Suter, 28. Jeff Petry, 29. Matt Niskanen, 30. Colton Parayko, 31. Oliver Ekman-Larsson, 32. Adam Fox, 33. Alexander Edler, 34. Aaron Ekblad, 35. Jaccob Slavin, 36. Ryan Graves, 37. Alec Martinez, 38. Jake Muzzin, 39. Duncan Keith, 40. Jared Spurgeon, 41. Jakob Chychrun, 42. Vince Dunn, 43. Darnell Nurse, 44. Jacob Trouba, 45. Mattias Ekholm, 46. Alex Goligoski, 47. Ryan Pulock, 48. Josh Morrissey, 49. Sami Vatanen, 50. Kevin Shattenkirk.