Open for Innovation
KNIME

Codeless Reinforcement Learning: Building a Gaming AI

Corey Weisinger

November 20, 2020

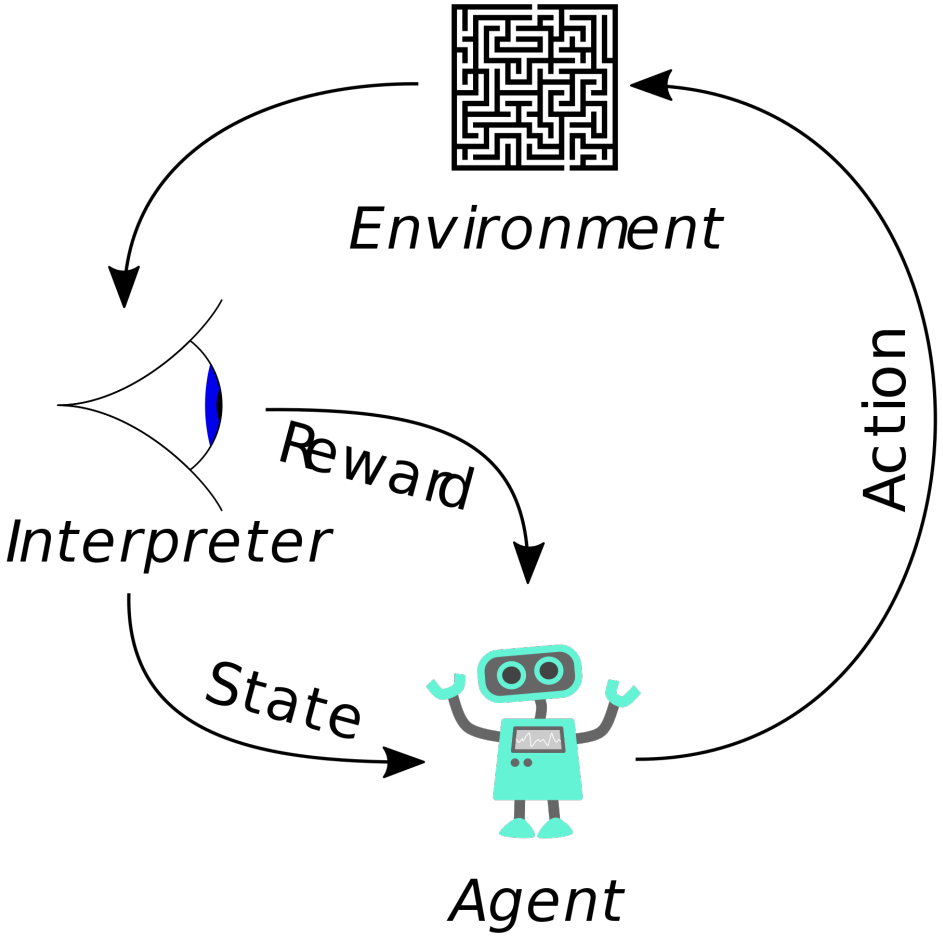
Goal and Agenda

The Goal for today is to provide high level intuitive and theoretical understandings of Reinforcement Learning, as well as to demonstrate them in a simple use case: a Tic-Tac-Toe playing AI in KNIME.

1. An overview of Reinforcement Learning and it's uses
2. Markov Decision Processes
3. Notation Introductions
4. Policy Functions
5. Reward Functions
6. Example Use Case and Implementation
7. Demo
8. Conclusions and Questions

Reinforcement Learning

- Reinforcement Learning has become a broad field, encompassing many frameworks.
- The goal is to design an Agent that can interpret its Environment and maximize its Reward.
- This Agent could be composed entirely, in part, or not at all, of a Machine Learning Model.



A Simple Agent

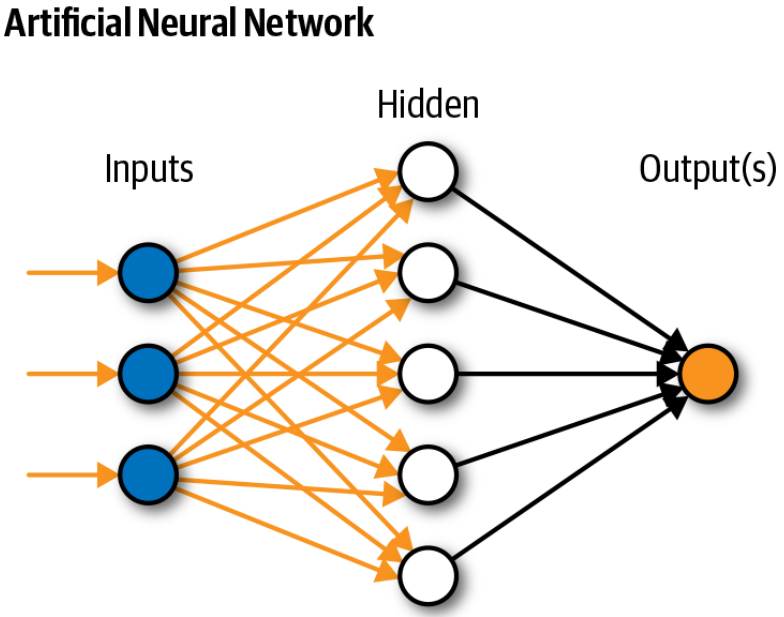
- Imagine an Agent designed to play Rock-Paper-Scissors. Further, let's say it can make it’s move after yours.

- In this case a look-up table is still very practical, we can even put it in half a slide!

Player’s Move	Agent’s Move
Rock	Paper
Paper	Scissors
Scissors	Rock

When a Look-Up Table Doesn’t Work.

- When we start thinking about more complicated environments for our Agent to live in, we need something more complicated than a look up table.
- This is the motivation for using Machine Learning.
- Helpful features when choosing a model.
  - Flexible inputs and outputs
  - Iterative training algorithm
- For this reason, Neural Networks are commonly used, and Decision Trees are rarely used.

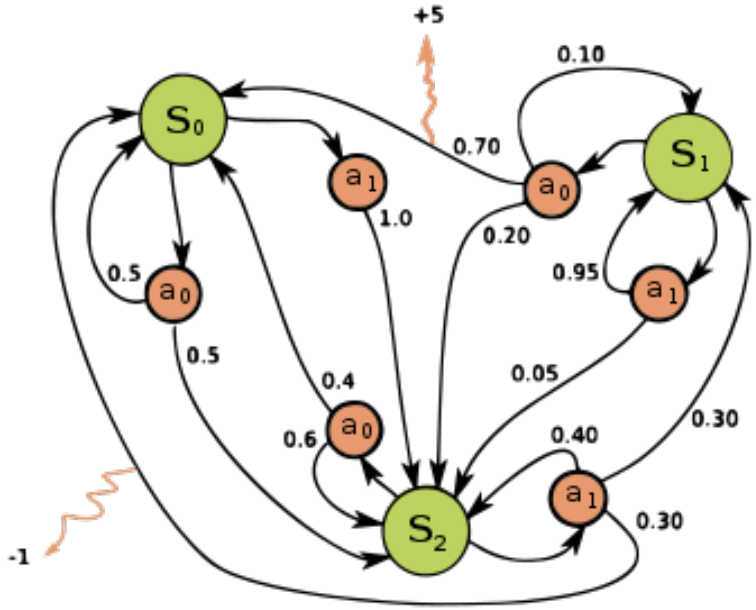


How Reinforcement Learning is Used

- Chemical Drug Discovery
  - Reinforcement Learning is often used to create chemical formulas/compounds with a desired set of physical properties, according to the following sequence of actions: Generate compound > test properties > update model > repeat.
- Traffic Light Control
  - Reinforcement Learning has made huge strides here as well. However, in this case, the partially observable nature of real-world traffic adds some layers of complexity.
- Robotics
  - This is likely the most famous of these use cases. We've seen many examples of robots learning to walk, pick up objects, or shake hands in recent years.

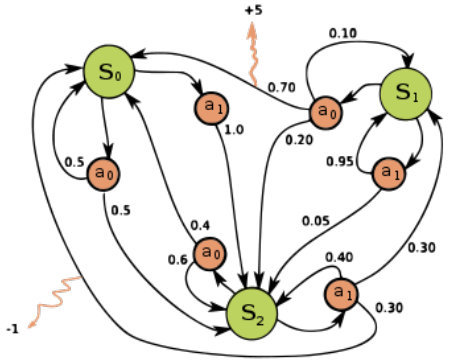
Markov Decision Process

- Reinforcement Learning is built on top of the idea of a Markov Decision Process
- We move around environment states, **s**, by taking actions, **a**
- These different actions have different probabilities of moving us to different environment states
- We now have a stochastic decision-making framework



Notation for Reinforcement Learning

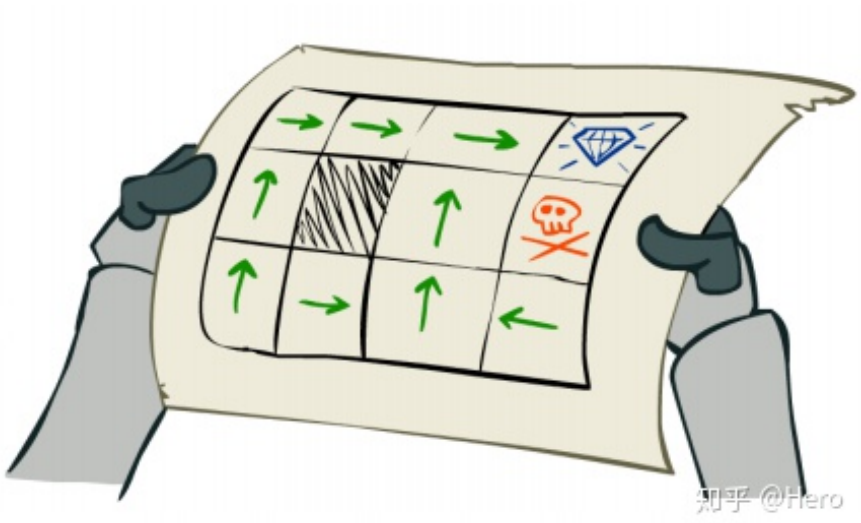
Notation	Meaning
a	Action taken
s	Environment State before Action
s'	Environment State after Action
$R_a(s,s')$	Reward for moving from State s to State s' after Action a
$P_a(s,s')$	Probability of moving from State s to State s' after Action a
A	Set of all possible Actions
S	Set of all possible Environment States
$\pi(s)$	Policy Function, defining what action to take at State s



Policy Functions: $\pi(s)$

- The Policy Function defines how your agent makes its choices
- Depending on the scenario this may be complex or simple

- $\pi(s)$ = move up
  - when navigating a grid
- $\pi(s)$ = hit me
  - when playing blackjack
- $\pi(s)$ = a; where a is a random element of A
  - a common starting point train an ML based function



Reward Functions

- We need a concept of “Reward” so our agent knows how to improve its Policy Function
- This function can change drastically by use case
- For example if your goal is to create stable chemical compounds a % stable or average lifespan may be sufficient
- In games with scores this is also easy, use the score
- Sometimes though we need to get creative, or even use multiple Rewards

Example Use Case – Tic-Tac-Toe

- Keras, Neural Net, based Agent
- Reward based on turns to win (or negatively turns to lose)
- Playable application in KNIME Webportal

Tic-Tac-Toe

- a KNIME Videogame -

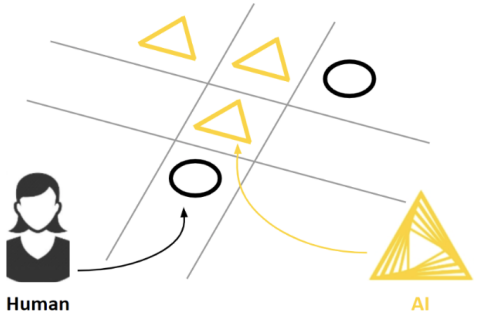
Powered by Reinforcemnt Learning.

Challenge the KNIME AI in a tick tack toe game!

- KNIME icons represent the moves of the AI player.
- Human icons represent your moves.
- To make a play select unclaimed cells and click "Next".

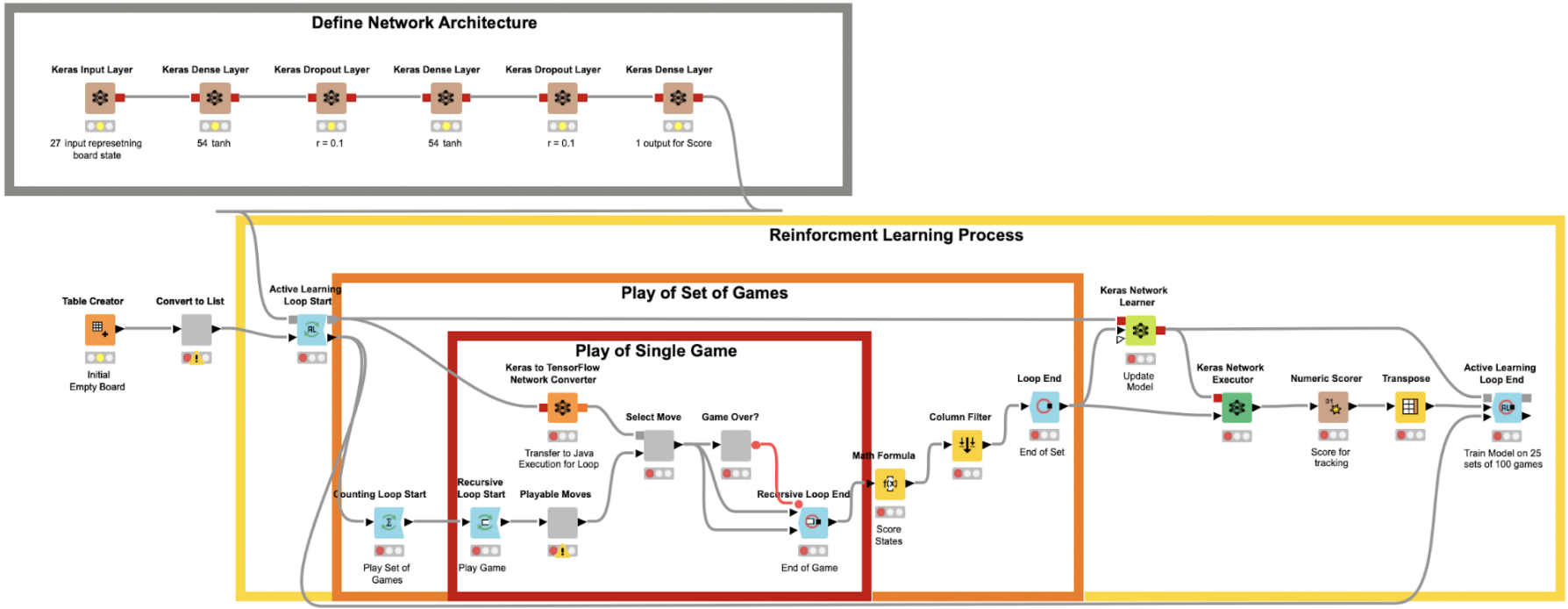
Select Difficulty: (•) Hard Mode ( ) Easy Mode

Select who should start the game: (•) KNIME AI ( ) Me!



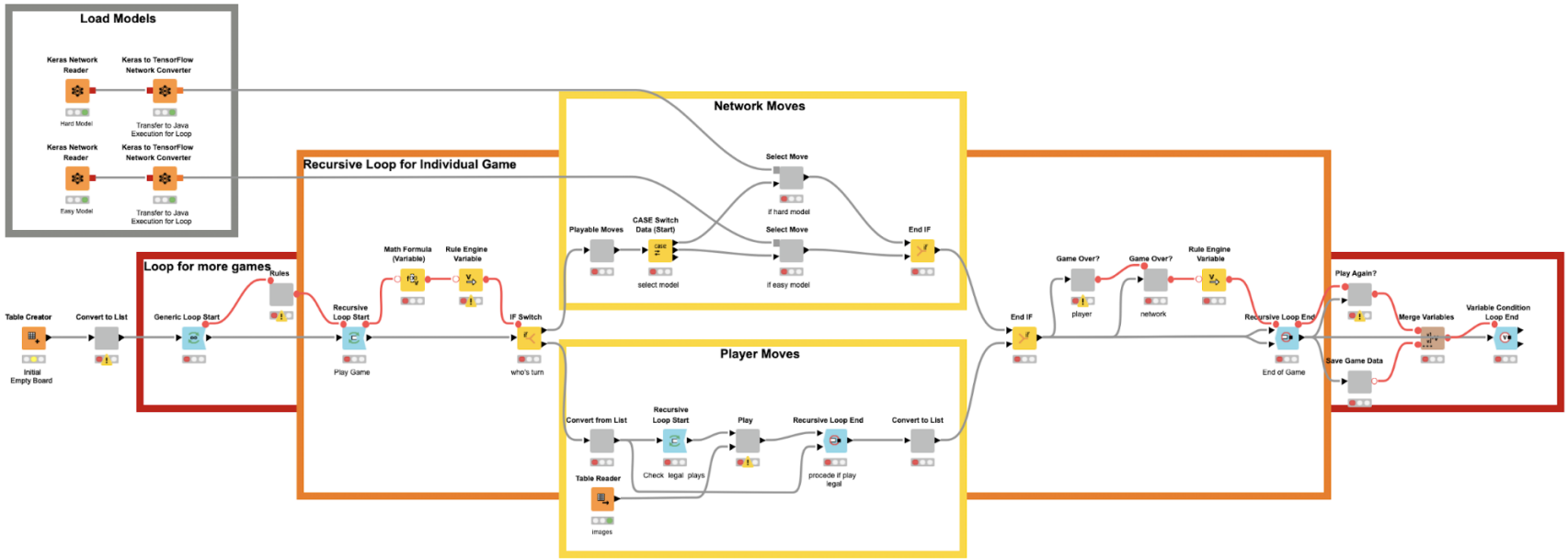
Implementation – Training the AI

- Define Architecture
- Play Games
- Update Model



Implementation – Playing the Game

- Load Trained Network
- Alternate AI vs Human Player
- Check for winner



Demo

Tic-Tac-Toe

- a KNIME Videogame -

Powered by Reinforcemnt Learning.

Challenge the KNIME AI in a tick tack toe game!

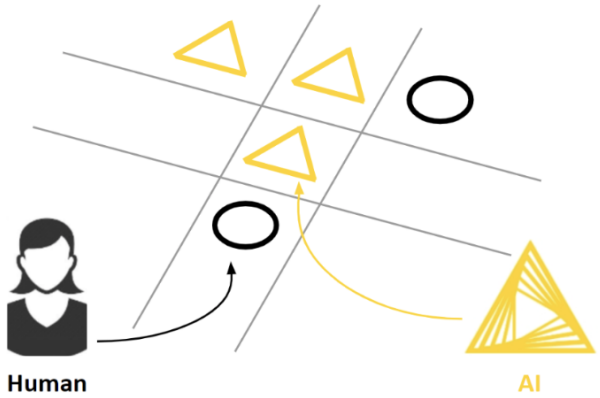
- KNIME icons represent the moves of the AI player.
- Human icons represent your moves.
- To make a play select unclaimed cells and click "Next".

Select Difficulty:

- (●) Hard Mode
- ( ) Easy Mode

Select who should start the game:

- (●) KNIME AI
- ( ) Me!

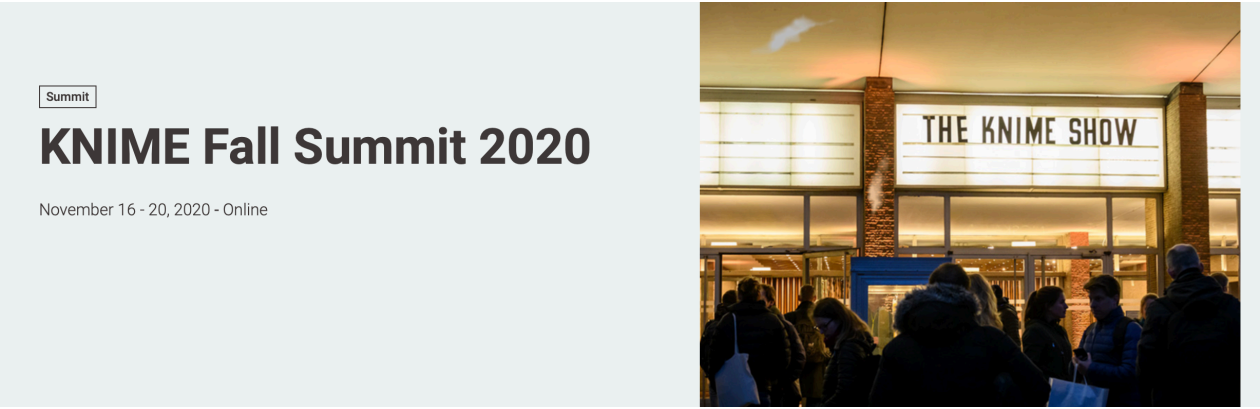


Conclusions

- Reinforcement Learning has many practical applications beyond the complex ones that dominate the news
- Reinforcement Learning is based on Markov Decision Process: a stochastic framework for modeling decisions
- The Policy Function, $\pi(s)$, is how our Agent makes it choices
- The Reward Function, $R_a(s, s')$, is how our Agent evaluates its performance
- Neural Networks are popular for their ability to model complex environment and for their iterative training algorithms
- GUI tools are becoming more and more robust and we can deploy simple Reinforcement Learning Agents directly in KNIME

Thank You! Questions?

- Book Coupon Code: **FALL-SUMMIT-WORKSHOP**
  - https://KNIME.com/knimepress
- KNIME Fall Summit:
  - https://www.knime.com/about/events/knime-fall-summit-2020
- Blog Post on content and use case:
  - https://www.knime.com/blog/an-introduction-to-reinforcement-learning
- Add me on LinkedIn:
  - https://www.linkedin.com/in/corey-weisinger-709525121/



Additional Reading

- Markov Decision Process
- Reinforcement Learning
- Deep Reinforcement Learning
- KNIME Transfer Learning Blog
- Tic-Tac-Toe on wiki
- Chemical Drug Discovery
- Bayesian Optimization
- Traffic Regulation
- Deep RL for Robotics
- KNIME Analytics Platform
- KNIME Server
- KNIME WebPortal
- KNIME and Keras
- KNIME Parameter Optimization
- KNIME Cheat Sheet on Machine Learning
- Tic-Tac-Toe Learning Workflow
- Tic-Tac-Toe Playing Workflow